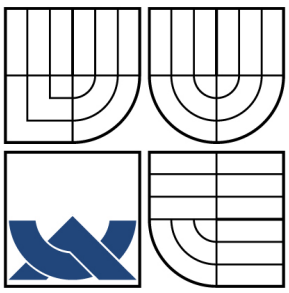
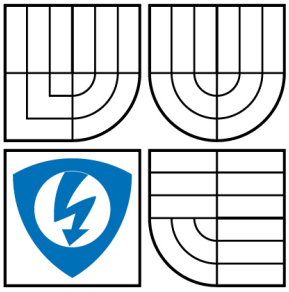


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

METODY NÁVRHŮ POČÍTAČOVÝCH SÍTÍ

DESIGN METHODS FOR COMPUTER NETWORKS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

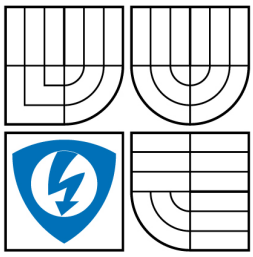
AUTOR PRÁCE — MARTIN BOŽEK

AUTHOR

VEDOUCÍ PRÁCE — Ing. JAN HAJNÝ

SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Martin Božek	***ID:***	98024
Ročník:	3	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Metody návrhů počítačových sítí

POKYNY PRO VYPRACOVÁNÍ:

Proveďte analýzu modelu TCP/IP především z bezpečnostního hlediska. Vyhledejte a popište prvky, které se ve středně velké infrastruktuře nejčastěji používají k zabezpečení sítě. Popište nejčastěji používané topologie a s ohledem na předchozí zjištění proveďte návrh sítě pro středně velkou společnost s velkým důrazem na bezpečnost. Prakticky realizujte klíčové prvky v návrhu. Zaměřte se především na hraniční prvky a jejich realizaci pomocí OpenSource softwaru. Proveďte zhodnocení a bezpečnostní analýzu.

DOPORUČENÁ LITERATURA:

[1] DOSTÁLEK, Libor, KABELOVÁ, Alena. Velký průvodce protokoly IP a systémem DNS. [s.l.] : [s.n.], 2002. 436 s.
[2] NORTHCUTT, Stephen. Bezpečnost počítačových sítí. [s.l.] : [s.n.], 2005. 592 s.

Termín zadání:	9.2.2009	***Termín odevzdání:***	2.6.2009

Vedoucí práce: Ing. Jan Hajný

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Bakalářská práce se zabývá problematikou vhodného návrhu počítačové sítě pro středně velkou infrastrukturu, jejím zabezpečením a následnou realizací směrovače s firewallem. Tento hraniční prvek bude schopný plnit základní úkony řízení síťového provozu mezi vnitřní a vnější sítí.

V této práci je stručně provedena analýza modelu OSI a podrobněji jsme se zaměřili na model TCP/IP. Dále naše práce obsahuje popis a nastiňuje význam nejčastěji používaných prvků v daném typu sítě a jejich přínos pro zabezpečení. Následuje popis tří nejběžněji používaných topologií v daném typu infrastruktury. Na základě zjištěných informací je vybrána a analyzována nejvhodnější topologie a to především z bezpečnostního hlediska. Výsledkem toeretické části práce je návrh modelu sítě.

V praktické části jsou některé nabyté poznatky otestovány a implementovány v reálném nasazení. Pro realizaci a otestování byla použita technologie virtualizace, pomocí které jsme nakonfigurovali router s firewallem, představující hraniční prvek, který odděluje vnitřní síť od vnějšího prostředí Internetu.

KLÍČOVÁ SLOVA

Směrovač, router, firewall, Linux, zabezpečení, síť

ABSTRACT

Bacherol‘s thesis deals with a topic of a suitable computer network design for a mid-sized infrastructure, its security and router and firewall realization. This boundary element is able to manage basic processes of network controlling between inner and outer network.

There is a brief analysis of OSI model but we focused our attention on TCP/IP model more closely. Our project includes description and outlines an importance of the most common elements of the particular network type and their contribution to the security. Description of the three most frequent topologies of the infrastructure type follows. On the basis of the found information, we have chosen and analysed the most suitable topology from the security point of view. The result of the theoretical part is a network model proposal.

In the practical part, some our findings are tested and implemented in a real setting. We used a technology of virtualization for the final realization and testing. Thanks to the virtualization used for realization and testing, we could configure the router and the firewall, which are boundary elements that separate the inner network from the Internet.

KEYWORDS

Router, firewall, Linux, security, network

BOŽEK, M. *Metody návrhů počítačových sítí.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 44 s. Vedoucí bakalářské práce Ing. Jan Hajný.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma "Metody návrhů počítačových sítí" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č.121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení §152 trestního zákona č. 140/1961 Sb.“

V Brně dne …… ……..

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Ing. Janu Hajnému za velmi užitečnou metodickou pomoc a cenné rady při zpracování diplomové práce.

V Brně dne …… ……..
(podpis autora)

Obsah

1 Úvod

V posledních několika letech se problematika zabezpečení sítí stala často diskutovaným fenoménem. V 90. letech proběhlo masové rozšíření informačních technologii do téměř všech oblastí lidské činnosti, které stále pokračuje. Právě tento rychlý rozmach ukázal, že ne všichni uživatelé IT jsou si vědomi rizik, které s sebou používání informačních technologií přináší. Docházelo (a stále dochází) k únikům citlivých informací, osobních údajů nebo firemních tajemství. Proto si je třeba uvědomit, že používání zabezpečení počítačových systémů není luxusem, ale nezbytnou nutností, ať už mluvíme o velké organizaci nebo domácnosti.

Technologie zabezpečení se neustále vyvíjí, ale techniky útoků a napadení s nimi drží krok a jejich množství má spíše vzestupnou tendenci. Proto je vhodné zabezpečení nikdy nepodceňovat a přiměřeně reagovat na nové trendy.

Útok na počítačový systém může mít mnoho podob – může se jednat o snahu dešifrovat citlivá data, fyzické napadení firemních serverů či počítačový virus. V této práci se nejprve zaměřujeme na návrh sítě pro organizaci střední velikosti, která bude schopná odolat typickým formám síťových útoků, včetně snahy proniknout do sítě a poškodit nebo odcizit citlivá firemní data. V praktické části práce se budeme zabývat realizací hraničního směrovače, který bude schopen plnit základní úkony pro směrování a filtrování síťového provozu mezi vnitřní sítí a Internetem.

Abychom mohli dosáhnout tohoto cíle, v první části práce provedeme analýzu modelu TCP/IP, který je základem dnešních sítí a Internetu. V další části práce se zaměříme na typické prvky využívané k zabezpečení sítě, metody zabezpečení a nejčastěji používané síťové topologie vhodné pro ochranu daného typu sítě. Na základě vyhodnocení těchto informací navrhneme vhodný model sítě, který bude plně využitelný pro středně velkou firmu a zároveň bude dostatečně a efektivně zabezpečený. Nabyté poznatky a postupy prakticky zhodnotíme a zároveň otestujeme v poslední části práce, kde pomocí virtualizace budeme simulovat router s firewallem, oddělující vnitřní síť od vnějšího prostředí Internetu.

1.1 Historie

Vznik a rozvoj počítačových sítí je neoddělitelně spojen s vývojem počítačů a výpočetní techniky. Prudký rozvoj výpočetní techniky začal již v padesátých letech dvacátého století, ovšem potřeba propojit počítače, tehdy o velikosti celých místností nebo sálů, přišla až téměř o dvacet let později. Tehdy počítače mohly mít uloženo relativně velké množství dat, ovšem jejich dostupnost a přenos byly problém. Tím začalo období propojování velkých počítačů a týkalo se především území Spojených států amerických, kde se vývoje v této oblasti účastnily vojenské a vládní organizace, univerzity a velké firmy. Výsledkem bylo vytvoření mnoha privátních sítí, ale vyvinuta byla i řada projektů, jako například síť americké armády ARPANET, která dala vzniknout protokolům TCP / IP a tato protokolová sada je i po více než třiceti letech základem přístupu k Internetu. Dalším významným projektem byla síť ALOHA havajské univerzity, ze které těžila technologie Ethernet a za zmínku stojí i technologie DECnet firmy Digital, která byla později zahrnuta do modelu OSI. V průběhu sedmdesátých let se stalo trendem postupné propojování akademických sítí a později i sítí ostatních, což dalo vzniknout Internetu.

V roce 1981 firma IBM uvedla na trh nový typ mikropočítače s názvem IBM PC, který svou otevřeností architektury, výkonností a variabilitou, způsobil menší revoluci. Do té doby byly počítače svou architekturou určeny především do větších firem nebo jiných komplexů. Po příchodu IBM PC počítače začaly být chápány jako nástroj, který mohl posloužit jednotlivci. Během pár let se PC rozšířily tak, že sálové počítače ve firmách začaly ustupovat a téměř každý úředník, manager nebo výzkumník měl na stole svůj osobní počítač, na němž měl uložena svoje data a prováděl výpočty. Se vzrůstající výpočetní silou osobních počítačů a rozvojem technologie, začalo docházet ke stejnému jevu, jako u počítačů sálových několik let předtím. Vyhledávání, dostupnost a přenos dat mezi osobními počítači byl problém a tak došlo na jejich propojování do lokálních sítí. [1]

Začal vývoj nových technologií, jehož základem byly již odzkoušené technologie a poznatky, ale orientoval se především na lokální sítě. Technologie Ethernet, ARCnet a Token Ring patřily k nejvýznamnějším. Díky masivnímu marketingu a široké podpoře firem zabývajících se výrobou integrovaných obvodů, zvítězila technologie Ethernet, přestože byla problémová. V lokálních sítích se Ethernet prosadil přibližně v osmdesáti procentech všech instalací.

Historii samotné protokolové sady TCP/IP můžeme shrnout následovně. Původním protokolem pro komunikaci v rámci sítě ARPANET byl NCP (Network Control Protocol), ale s postupem času a vývojem pokročilejších technologií začal být NCP nahrazován výrazně propracovanějším standardem vyšší úrovně označovaným TCP/IP. Tento standart vznikl jako výsledek projektu agentury DARPA, jehož cílem bylo zkoumat techniky a technologie propojování paketových sítí různých typů. Systém sítí navržený v rámci tohoto projektu vešel ve známost pod jménem Internet.

2 Síťové protokoly

V počátcích budování sítí byly sítě od různých výrobců mezi sebou nekompatibilní. Z těchto důvodů vznikla snaha komunikaci v počítačových sítí standartizovat, aby i při použití různého hardwaru byla zajištěna spolehlivá a bezchybná komunikace. Mezinárodní organizace pro normalizaci ISO, založila v roce 1977 vlastní výbor, pod názvem OSI (Open system interconnection) a o dva roky později vytvořila RM OSI (referenční model pro propojování otevřených systémů). Přestože byl tento model určen pro využití v sítích WAN (Wide Area Network – rozsáhlá síť), je možné jej využít i v sítích lokálních.

2.1 Referenční model OSI

Referenční model OSI tvoří sedm vrstev a na jednotlivých vrstvách specifikuje protokoly a jejich vzájemnou spolupráci. Při komunikaci v síti jednotlivé vrstvy mezi sebou komunikují jen s odpovídající vrstvou, např. transportní vrstva s transportní atd. [2]

Fyzická vrstva – nejnižší vrstva modelu, která jednotlivým bitům umožňuje přenos komunikačním kanálem, bez ohledu na jejich význam. Definuje fyzické signály a jejich vlastnosti, které zastupují logickou 1 a logickou 0. Dále vrstva stanovuje vlastnosti přenosového média, přenosovou rychlost, tvary konektorů atd. Na této vrstvě pracuje repeater a hub.

Linková vrstva – jejím cílem je zajistit bezchybnou výměnu dat mezi sousedními počítači popř. výměnu dat v rámci lokální sítě. Tvoří rámce obsahující kromě vlastních přenášených

dat i informace o adresování, zabezpečení proti chybám přenosu a údaj pro rozpoznání počátku rámce. Na této vrstvě pracuje bridge a switch.

Síťová vrstva – zabezpečuje adresaci a směrování dat v rozlehlé síti od zdroje k cíli. Základní jednotkou přenosu je síťový paket, který se balí do datového rámce. Paket obsahuje záhlaví a datové pole. Na této vrstvě pracuje router.

Transportní vrstva – zajišťuje vlastní přenos dat. Zabezpečuje, aby se zpráva dostala k příjemci celá a správně. Vyrovnáná odlišnou kvalitu přenosových sítí.

Relační vrstva – cílem této vrstvy je navázání relací mezi koncovými stanicemi**.** Zabezpečuje synchronizaci transakcí, práva, hesla, omezení, korektní uzavírání souborů...

Prezentační vrstva – zajišťuje zabezpečení dat a převod formátu dat do univerzální podoby. Odpovídá tedy za způsoby kódování, komprimaci, kryptografii, integritu...

Aplikační vrstva – předepisuje v jakém formátu a jakým způsobem mají bát data předávána a přebírána od aplikací. Je představována např. FTP, Telnet a SMTP. Na této vrstvě pracuje gateway.

<table>
<tr><th>OSI</th><th>TCP / IP</th><th colspan="8">Aplikace a protokoly</th></tr>
<tr><td>7. aplikační</td><td rowspan="3">Aplikační vrstva</td><td rowspan="3">Telnet</td><td rowspan="3">FTP</td><td rowspan="3">TFTP</td><td rowspan="3">SMTP</td><td rowspan="3">RIP</td><td rowspan="3">DNS</td><td rowspan="3">Ostatní</td></tr>
<tr><td>6. prezentační</td></tr>
<tr><td>5. relační</td></tr>
<tr><td>4. transportní</td><td>Transportní vrstva</td><td colspan="5">TCP</td><td colspan="3">UDP</td></tr>
<tr><td rowspan="2">3. síťová</td><td rowspan="2">Síťová vrstva</td><td colspan="4" rowspan="2">IP</td><td>ICMP</td><td colspan="3"></td></tr>
<tr><td colspan="2">ARP</td><td>RARP</td></tr>
<tr><td>2. linková</td><td rowspan="2">Vrstva síťového rozhraní</td><td colspan="8" rowspan="2">Token ring Ethernet Jiné typy protokolů</td></tr>
<tr><td>1. fyzická</td></tr>
</table>

Obr. č. 1 – přehled architektury TCP / IP

2.2 Protokol TCP / IP

Z modelu ISO OSI vychází i množina protokolů TCP/IP, který původně vznikl jako komunikační protokol ministerstva obrany USA, jehož účelem bylo sjednotit počítačovou komunikaci v rámci sítě ARPANET. Přestože byl navrhován pro operační systém UNIX,

v dnešní době je zahrnut ve všech operačních systémech. Je využit především ke komunikaci v rámci Internetu, což z něj činí standart celosvětového významu. [2]

2.2.1 Vztah ISO OSI a TCP/IP

Obě skupiny mají svou vlastní definici vrstev i protokolů jednotlivých vrstev. V praxi je třeba využívat komunikační zařízení vyhovující ISO OSI pro přenos IP-paketů nebo například naopak realizovat služby podle ISO OSI přes Internet. Na Obr. č. 1 vidíme rozdílné pojetí vrstev obou protokolů.[2]

2.2.2 Vrstvy TCP/IP

Protokol TCP/IP je členěný do čtyř vrstev (viz. Obr. č. 1), nyní si je stručně popíšeme.

Vrstva síťového rozhraní - nejnižší vrstva umožňuje přístup k fyzickému přenosovému médiu. Je specifická pro každou síť v závislosti na její implementaci. Příklady sítí: Ethernet, Token ring, FDDI, X.25, SMDS.[2]

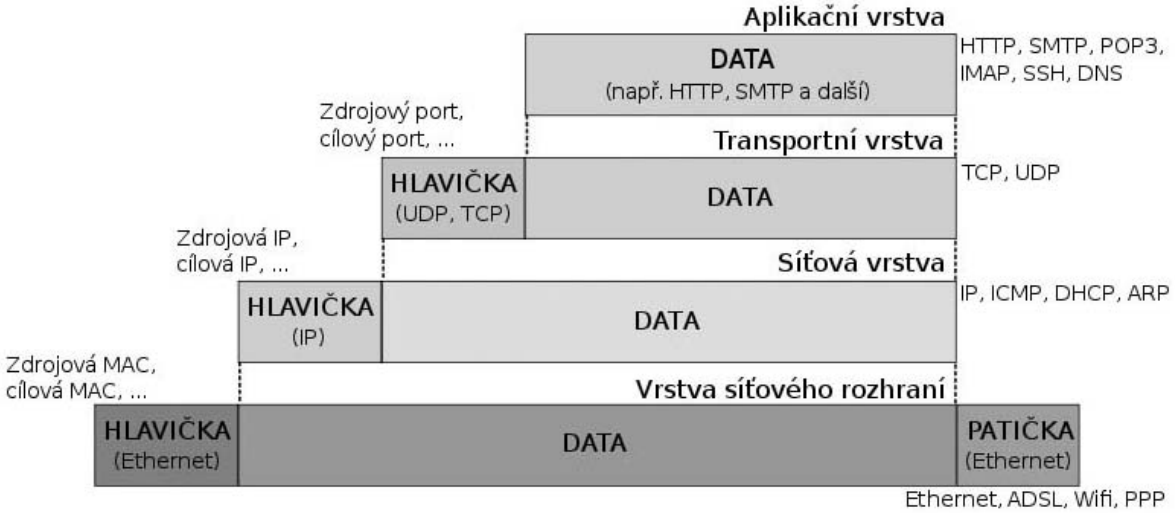
Síťová vrstva - Vrstva zajišťuje především síťovou adresaci, směrování a předávání datagramů. Protokoly: IP, ARP, RARP, ICMP, IGMP, IGRP, IPSEC. Je implementována ve všech prvcích sítě - směrovačích i koncových zařízeních.[2]

Transportní vrstva - Transportní vrstva je implementována až v koncových zařízeních (počítačích) a umožňuje proto přizpůsobit chování sítě potřebám aplikace. Poskytuje spojované (protokol TCP, spolehlivý) či nespojované (UDP, nespolehlivý) transportní služby. [2]

Aplikační vrstva - Vrstva aplikací. Jsou to programy (procesy), které využívají přenosu dat po síti ke konkrétním službám pro uživatele. Příklady: Telnet, FTP, HTTP, DHCP, DNS.[2]

Aplikační protokoly používají vždy jednu ze dvou základních služeb transportní vrstvy: TCP nebo UDP, případně obě dvě (např. DNS). Pro rozlišení aplikačních protokolů se používají tzv. porty, což jsou domluvená číselná označení aplikací. Každé síťové spojení aplikace je jednoznačně určeno číslem portu a transportním protokolem (a samozřejmě adresou počítače).[2]

Pro lepší pochopení struktury práce protokolu TCP / IP s daty viz. Obr. č. 2 [2]



Obr. č. 2 – zapouzdření dat v síti TCP/IP

2.2.3. IP protokol

2.2.3.1. Funkce protokolu

Data se v IP síti posílají po blocích nazývaných datagramy. Jednotlivé datagramy putují sítí zcela nezávisle, na začátku komunikace není potřeba navazovat spojení či jinak „připravovat cestu“ datům, přestože spolu třeba příslušné stroje nikdy předtím nekomunikovaly. [2] [3]

Často se můžeme setkat se zaměňováním pojmu datagram a paket. Datagram je druh paketu nespolehlivého přenosu dat, zatímco paket je obecným označením zprávy ve formátu paketu.

IP v doručování datagramů poskytuje nespolehlivou službu, označuje se také jako best effort – „nejlepší úsilí“; tj. všechny stroje na trase se datagram snaží podle svých možností poslat blíže k cíli, ale nezaručují prakticky nic. Datagram vůbec nemusí dorazit, může být naopak doručen několikrát a neručí se ani za pořadí doručených paketů. Pokud aplikace potřebuje spolehlivost, je potřeba ji implementovat v jiné vrstvě síťové architektury, typicky protokoly bezprostředně nad IP (viz. kapitola 2.2.4 TCP protokol). Pokud by síť často ztrácela pakety, měnila jejich pořadí nebo je poškozovala, výkon sítě pozorovaný uživatelem by byl malý. Na druhou stranu příležitostná chyba nemívá pozorovatelný efekt. Navíc se obvykle používá vyšší vrstva, která ji automaticky opraví [2] [3].

Data jsou od odesílatele k příjemci dopravována (směrována) přes směrovače (router). Na cestě od odesílatele k příjemci se může vyskytnout celá řada směrovačů. Každý směrovač řeší samostatně směrování k následujícímu směrovači. Data jsou tak předávána od směrovače k směrovači. Z angličtiny se počeštil v tomto kontextu termín následující hop (next hop), jako následující uzel kam se data předávají. Hopem se rozumí buď následující směrovač nebo cílový stroj. [2]

IP-protokol je tvořen několika dílčími protokoly:

- Vlastním protokolem IP.
- Služebním protokolem ICMP sloužícímu zejména k signalizaci mimořádných stavů.
- Služebním protokolem IGMP sloužícímu pro dopravu adresných oběžníků.
- Služební protokoly ARP a RARP, které jsou často vyčleňovány jako samostatné na IP nezávislé protokoly, protože jejich rámce nejsou předcházeny IP-záhlavím.

2.2.3.2. IP-datagram

	Bity 0–3	4–7	8–15	16–18	19–31
0	Verze	Délka hlavičky	Typ služby	Celková délka	
32	Identifikace			Příznaky	Posun fragmentu
64	Životnost (TTL)		Protokol	Kontrolní součet hlavičky	
96	Adresa odesilatele				
128	Cílová adresa				
160	Volby (volitelná délka)				
…	Data				

Obr. č. 3 - Struktura IP-datagramu IP verze 4 (IPv4) [2]

- **Verze**: Verze IP.
- **Délka hlavičky**: Délka hlavičky vyjádřená ve čtyřbajtových slovech.
- **Typ služby (Type of Service, TOS)**: Obsahuje identifikaci pro mechanismy, které zajišťují služby s definovanou kvalitou (QoS).
- **Celková délka**: Délka datagramu vyjádřená v bajtech.
- **Identifikace**: Na straně odesilatele je přidělen každému odchozímu paketu jednoznačný identifikátor.
- **Příznaky**: Slouží k řízení fragmentace.

- **Posun fragmentu:** určuje kde v původním datagramu začíná tento fragment.
- **Životnost (Time To Live, TTL)**: Je ochranou proti zacyklení. Po průchodu směrovačem je tato hodnota vždy zmenšena o jedničku. Jestliže dosáhne hodnoty 0 je zahozen, protože vypršela jeho životnost.
- **Protokol**: Udává, kterému protokolu z vyšších vrstev se mají při doručení předat data.
- **Kontrolní součet**: Kontroluje jestli nedošlo k poškození dat. Je vypočítán z hlavičky datagramu a pokud neodpovídá, datagram je zahozen.
- **Adresa odesilatele**: IP adresa počítače, z kterého byl datagram vyslán.
- **Cílová adresa**: IP adresa stanice, pro kterou je datagram určen.
- **Volby**: Rozšiřující doplňkové informace nebo požadavky.

Za výše popsanou hlavičkou následuje případná výplň, která její délku dorovná na násobek čtyř bajtů (pokud jsou volby kratší, délka hlavičky se uvádí ve čtyřbajtových slovech). Za ní jsou pak umístěna samotná nesená data. [2]

2.2.3.3 Protokol ICMP

ICMP protokol (Internet Control Message Protocol) je jeden z nejdůležitějších protokolů ze sady Internet Protocol. Používají ho operační systémy počítačů v síti pro odesílání chybových zpráv - například pro oznámení, že požadovaná služba není dostupná nebo že potřebný počítač nebo router není dosažitelný. [2]

ICMP se svým účelem liší od TCP a UDP protokolů tím, že se obvykle nepoužívá síťovými aplikacemi přímo. Jedinou výjimkou je nástroj ping, který posílá ICMP zprávy „Echo Request“ (a očekává příjem zprávy „Echo Reply“) aby určil, zda je cílový počítač dosažitelný a jak dlouho paketům trvá, než se dostanou k cíli a zpět. [2]

Nejpoužívanější ICMP datagramy[2]:

- Echo Request - požadavek na odpověď, každý prvek v síti pracující na IP vrstvě by na tuto výzvu měl reagovat. Často to z různých důvodů není dodržováno.
- Echo Reply - odpověď na požadavek.
- Destination Unreachable - informace o nedostupnosti cíle, obsahuje další upřesňující informaci, např.: Net Unreachable, Host Unreachable, Port Unreachable.
- Redirect - přesměrování, používá se především pokud ze sítě vede k cíli lepší cesta než přes defaultní bránu.

- Time Exceeded - vypršel časový limit.

2.2.3.4 Protokol ARP a RARP

Address Resolution Protocol (ARP) se používá k získání ethernetové (MAC) adresy sousedního stroje z jeho IP adresy. Používá se v situaci, kdy je třeba odeslat IP datagram na adresu ležící ve stejné podsíti jako odesilatel. Data se tedy mají poslat přímo adresátovi, u něhož však odesilatel zná pouze IP adresu. Pro odeslání prostřednictvím např. Ethernetu ale potřebuje znát cílovou ethernetovou adresu. [2]

Proto vysílající strana odešle ARP dotaz (ARP request) obsahující hledanou IP adresu a údaje o sobě (vlastní IP adresu a MAC adresu). Tento dotaz se posílá linkovým broadcastem – na MAC adresu identifikující všechny účastníky dané lokální sítě . ARP dotaz nepřekročí hranice dané podsítě, ale všechna k ní připojená zařízení dotaz obdrží a jako optimalizační krok si zapíší údaje o jeho odesilateli do své ARP cache. Vlastník hledané IP adresy pak odešle tazateli ARP odpověď (ARP reply) obsahující vlastní IP adresu a MAC adresu. Tu si tazatel zapíše do ARP cache a může odeslat datagram. [2]

Reverse Address Resolution Protocol (RARP) se používá k získání vlastní IP adresy počítače při znalosti MAC adresy.

Vysílající strana vyšle RARP dotaz (RARP request) obsahující vlastní MAC adresu. Dotaz se posílá na MAC broadcast, tedy všem počítačům v dané fyzické síti. V ní by se měl nacházet RARP server opatřený tabulkou obsahující IP adresy příslušející jednotlivým MAC adresám. Server prohlédne tabulku a pokud v ní najde MAC adresu tazatele, pošle mu zpět RARP odpověď (RARP reply) s IP adresou, kterou si má nastavit. [2]

RARP umožňuje centrální správu IP adres, trpí však dvěma významnými nedostatky:

- Dotaz se posílá na fyzickou (MAC) broadcastovou adresu, nepřekročí tedy hranice fyzické sítě. V důsledku toho nelze mít v rozsáhlejší síti složené z několika podsítí jeden společný RARP server. [2]
- Předává pouze IP adresu. Stanice však ke svému síťovému životu potřebuje více informací (masku podsítě, implicitní bránu, adresu DNS serveru). Tyto informace nelze přenášet prostřednictvím RARP. [2]

Důsledkem těchto nevýhod je, že se RARP prakticky nepoužívá. Pro automatickou konfiguraci stanic se častěji nasazují lepší protokoly DHCP nebo BOOTP. [2]

2.2.4 TCP protokol

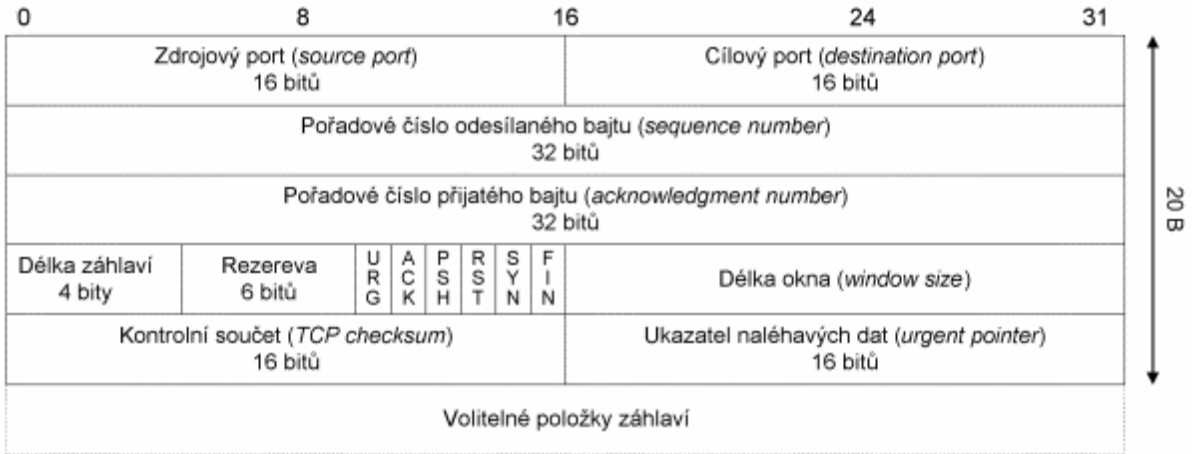
2.2.4.1 Funkce TCP protokolu

Použitím TCP mohou aplikace na počítačích propojených do sítě vytvořit mezi sebou spojení, přes které mohou přenášet data. Spojení je plně duplexní (data se přenášejí současně na sobě nezávisle oběma směry). Přenášené bajty jsou číslovány. Ztracená nebo poškozená data jsou znovu vyžádána a integrita přenášených dat je zabezpečena kontrolním součtem. TCP také rozlišuje data pro vícenásobné, současně běžící aplikace (například webový server a emailový server) běžící na stejném počítači. TCP podporuje mnoho důležitých aplikačních protokolů a aplikací, včetně WWW, e-mailu a SSH. [2] [4]

Cílová aplikace je v Internetu adresována (jednoznačně určena) IP-adresou, číslem portu a použitým protokolem (TCP nebo UDP). Protokol IP dopraví IP-datagram na konkrétní počítač. Na tomto počítači běží jednotlivé aplikace. Podle čísla cílového portu operační systém pozná, které aplikaci má TCP-segment doručit.

2.2.4.2 TCP segment

TCP rozděluje proud bajtů do přiměřeně velkých segmentů. Velikost segmentů je určena parametrem MTU (maximum transmission unit - maximální přenosová jednotka) linkové vrstvy sítě, ke které je počítač připojen.) TCP pak předá takto vzniklé pakety IP protokolu k přepravě Internetem do TCP modulu na druhé straně TCP spojení. TCP ověří, že se pakety neztratily tím, že každému paketu přidělil číslo sekvence, které se také použije k ověření, že data byla přijata ve správném pořadí. [2]



Obr. č. 4 - TCP segment [2]

- **Zdrojový port** - port odesilatele TCP segmentu. Jednoznačná identifikace v daném okamžiku spojení v Internetu je určena zdrojovým portem, cílovým portem, zdrojovou IP-adresou, cílovou IP-adresou a protokolem (TCP).
- **Cílový port** - port adresáta TCP segmentu.
- **Pořadové číslo odesílaného bajtu -** pořadové číslo TCP segmentu ve směru dat od odesílatele k příjemci.
- **Pořadové číslo přijatého bajtu -** číslo následujícího bajtu, který příjemce očekává k příjmu.
- **Délka záhlaví -** udává v násobcích 32 bitů délku záhlaví TCP segmentu.
- **Délka okna** - přírůstek pořadového čísla přijímaného bajtu, jenž bude příjemcem akceptován.
- **Ukazatel naléhavých dat –** lze použít jen v případě nastavení příznaku URG. Ukazatel se přičte k pořadovému číslu odesílaného bajtu a tím určí konec části naléhavých dat. Jeho použitím odesílatel značí, že se mají tato data zpracovat s vyšší prioritou.

V poli příznaků mohou být nastaveny následující příznaky[2]:

- **URG** – TCP segment nese naléhavá data.
- **ACK** – TCP segment má platné pole “Pořadové číslo přijatého bajtu” (nastaven ve všech segmentech, kromě prvního segmentu, kterým klient navazuje spojení).
- **PSH** – Zpravidla se používá k signalizaci, že TCP segment nese aplikační data, příjemce má tato data předávat aplikaci. Použití tohoto příznaku není ustáleno.
- **RST –** Odmítnutí TCP spojení.
- **SYN** – Odesilatel začíná s novou sekvencí číslování
- **FIN** – Odesilatel ukončil odesílání dat.

2.2.5 UDP protokol

2.2.5.1 Funkce UDP protokolu

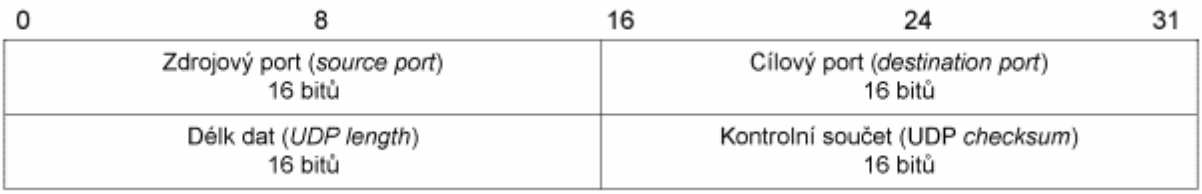
Protokol UDP nedává záruky na datagramy, které přenáší mezi počítači v síti. Bývá také označován jako nespolehlivý. Na rozdíl od protokolu TCP nezaručuje, jestli se přenášený

datagram neztratí, jestli se nezmění pořadí doručených datagramů nebo jestli se některý datagram nedoručí víckrát. [2] [5]

Protokol UDP je vhodný pro účely, které vyžadují jednoduchost nebo pro aplikace pracující systémem otázka-odpověď (např. DNS). Jeho bezstavovost je užitečná pro servery, které obsluhují mnoho klientů nebo pro nasazení, kde se počítá se ztrátami datagramů a není vhodné, aby se ztrácel čas novým odesíláním (starých) nedoručených zpráv (např. VoIP, online hry).

Kvůli chybějícím zárukám se UDP aplikace musí smířit s nějakými ztrátami, chybami nebo duplikacemi. Některé aplikace (jako třeba TFTP) mohou podle potřeby přidávat jednoduchý mechanismus spolehlivosti do aplikační vrstvy. Aplikace používající UDP naštěstí nejčastěji opravný mechanismus nepotřebují, a dokonce jím mohou být zdržovány. Pokud aplikace vyžaduje vysoký stupeň spolehlivosti, může se místo něj použít TCP nebo opravné kódy.

2.2.5.1 UDP datagram



Obr. č. 5 – Struktura UDP datagramu [2]

UDP hlavička se skládá jen ze 4 políček, z nichž 2 jsou volitelná. Políčka zdrojového a cílového portu jsou šestnáctibitová a identifikují odesílající a přijímající proces. Protože UDP je bezestavový a odesilatel nemusí vyžadovat odpověď, zdrojový port je volitelný. Pokud se nepoužije, zdrojový port by měl být nastaven na nulu. Po číslech portů následuje povinná délka UDP paketu včetně dat. Minimální hodnota je 8 bajtů. Zbývající políčko hlavičky je šestnáctibitový kontrolní součet pokrývající hlavičku i data. Tento součet je možné vynechat, ale v praxi se téměř vždy používá. [2] [5]

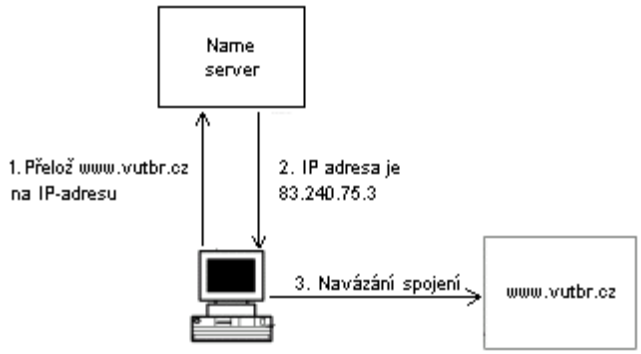
2.2.5 Domain name system (DNS)

DNS je systém doménových jmen, který je realizován servery DNS a protokolem stejného jména, kterým si vyměňují informace. Příčinou vzniku a hlavním úkolem jsou vzájemné převody doménových jmen a IP adres uzlů sítě. Později ale přibral další funkce (např. pro

elektronickou poštu či IP telefonii) a v podstatě dnes slouží jako distribuovaná databáze síťových informací. Záznamy, které jsou v databázy uchovávány jsou například:

- **NS** (name server record) – pro danou doménu udává jméno autoritavního DNS serveru.
- **A** (address record) – udává IPv4 adresu přidělenou danému jménu.
- **MX** (mail exchange record) – pro danou doménu udává adresu a stupeň priority serveru přijímajícím elektronickou poštu.
- **CNAME** (canonical name record) – pro již zavedené jméno označuje alias nebo-li přezdívku.

Tato celosvětově distribuovaná databáze má své jednotlivé části rozmístěné na tzv. name serverech. [2]



Obr. č. 6 – Přeložení jména na IP adresu před navázáním spojení

Prostor doménových jmen tvoří strom a každý uzel tohoto stromu obsahuje informace o části jména (doméně), které je mu přiděleno a odkazy na své podřízené domény. Kořenem stromu je tzv. kořenová doména, která se zapisuje jako samotná tečka. Pod ní se v hierarchii nacházejí tzv. domény nejvyšší úrovně (Top-Level Domain, TLD). Ty jsou buď tematické (např. com pro komerci) nebo státní (cz pro Českou republiku atd.). [2]

Strom lze administrativně rozdělit do zón, které spravují jednotliví správci (organizace nebo i soukromé osoby), přičemž taková zóna obsahuje autoritativní informace o spravovaných doménách. Tyto informace jsou poskytovány autoritativním DNS serverem.

Výhoda tohoto uspořádání spočívá v možnosti zónu rozdělit a správu její části svěřit někomu dalšímu. Nově vzniklá zóna se tak stane autoritativní pro přidělený jmenný prostor. Právě možnost delegování pravomocí a distribuovaná správa tvoří klíčové vlastnosti DNS a jsou velmi podstatné pro jeho úspěch. Ve vyšších patrech doménové hierarchie platí, že zóna

typicky obsahuje jednu doménu. Koncové zóny přidělené organizacím připojeným k Internetu pak někdy obsahují několik domén – například doména neco.cz a její poddomény jako např. nekde.neco.cz mohou být obsaženy v jedné zóně a obhospodařovány stejným serverem. [2]

3 Síťové prvky sloužící k zabezpečení sítě

3.1 Bezpečnostní politika středně velké infrastruktury

Připojení privátní (podnikové) sítě do veřejné sítě Internet znamená propojení této sítě s desítkami tisíců neznámých sítí se všemi jejich uživateli. Z této skutečnosti vyplývají značná bezpečnostní rizika. Je zapotřebí řešit zásadní problémy a to zejména:

- Jak ochránit důvěrné informace od subjektů, které nemají explicitní potřebu je mít k dispozici.
- Jak ochránit síť a její zdroje před útočníky a nepředvídanými událostmi přicházejícími zvnějšku.

Vzhledem k tomu, že se informační technologie rozvíjí a neustále mění, objevují se i nové hrozby a způsoby zneužití slabých míst. Proto se nikdy nedá zaručit, že je firma stoprocentně zabezpečená. Pomocí tří základních kroků se mohou přiměřeně podniková data, softwarové aplikace a každodenní provoz ochránit.

1. První krok představuje stanovení bezpečnostní politiky - stručně a obecně se popíše v jakých stavech se síť může/nemůže nacházet a co by v síti nemělo nikdy nastat.
2. V dalším kroku se stanoví model rizik – určí se zdroje rizik, které by reálně mohly nastat a navrhne se model zabezpečení, který tato rizika dokáže pokrýt s ohledem na jeho reálnou implementaci.
3. V posledním kroku se implementuje stanovená politika pomocí bezpečnostních mechanizmů.

Zásady zabezpečení by kromě jiného měly řešit:

- Náležité používání firemních systémů e-mailu a rychlého posílání zpráv.

- Ochranu provozních dat (například informací o zaměstnancích, zákaznících nebo účetnictví) a dalších citlivých informací.
- Postupy reakce na bezpečnostní hrozby, pokusy o narušení, ztrátu dat, poruchy systémů nebo sítě.
- Náležitou péči a používání protokolů a systémů ověřování (uživatelských jmen a hesel).

3.2 Přehled prvků

Přehled a popis prvků, které se nejčastěji používají v rámci obvodu sítě. [6]

Hraniční směrovač

Směrovač nebo-li router řídí provoz směřující do sítě, ven ze sítě i uvnitř firemní sítě. Hraniční směrovač je tedy poslední námi řízený směrovač před vstupem do Internetu. Proudí přes něj celý internetový provoz organizace. Svým prvotním a závěrečným filtrováním většinou slouží jako první i poslední obranná linie sítě před útoky.

Firewall

Obecně slouží k zabezpečování a řízení síťového provozu mezi sítěmi s odlišnou úrovní zabezpečení a důvěryhodnosti. Definuje pravidla pro komunikaci mezi sítěmi, které od sebe odděluje. Má významnou roli v zabezpečení sítě, proto jsem firewallům a jejich úloze věnoval kapitolu 3.3.

Detekční systém IDS

Systém detekce vniknutí IDS je svým způsobem poplašný systém, který vyhledává nežádoucí události a hlásí je. Detekční systém se může skládat z mnoha senzorů, které jsou umístěny na strategicky významných bodech sítě. Rozdělují se na dva typy – síťové IDS a hostitelské IDS.

Proxy server

Stará se o optimalizaci a ochranu síťového provozu a klientů sítě. V podstatě slouží jako prostředník mezi klientem a cílovou stanicí (serverem). Překládá požadavky klientů a vůči cílovému počítači vystupuje jako klient. Přijatou odpověď následně odesílá zpět na klienta.

Může se jednat jak o specializovaný hardware, tak o software provozovaný na běžném počítači.

Virtuální privátní síť (VPN)

VPN slouží k propojení několika počítačů na různých místech Internetu do jediné virtuální počítačové sítě. Přestože počítače mohou být v naprosto fyzicky nezávislých sítích na různých místech, prostřednictvím virtuální privátní sítě mezi sebou mohou komunikovat, jako by byly spojeny do jednoho síťovém segmentu.

Pomocí VPN lze zajistit například připojení firemních notebooků kdekoliv na Internetu do firemního Intranetu (vnitřní firemní sítě). K propojení je třeba VPN server, který má přístup na Internet i Intranet (může sloužit pouze pro jednoho klienta nebo sloužit jako hub a přijímat spojení od více klientů) a VPN klient, který se přes Internet připojí k serveru a prostřednictvím něj pak do Intranetu. VPN server pak plní v podstatě funkci síťové brány.

Zobecněním VPN je síťové tunelování, kdy se prostřednictvím standardního síťového spojení vytvoří virtuální linka mezi dvěma počítači, v rámci které pak lze navázat další síťová spojení.

Demilitarizovaná zóna (DMZ)

Demilitarizovaná zóna je speciálním typem sítě používaným ke zvýšení bezpečnosti komunikace s vnějším prostředím (Internetem). Používá se tehdy, když je zapotřebí oddělit komunikaci s vnějším prostředím od komunikace lokálních sítí. Tedy tam, kde je nežádoucí, aby komunikace počítačů/serverů s vnějšími sítěmi byla spojena s komunikací lokálních sítí. Často takovéto segmenty sítě (DMZ) používají veřejné IP adresy, které by se v lokální síti vyskytovat neměly. Obvykle je tohoto řešení využito k oddělení serverů od lokální sítě. Díky tomu v případě napadení serveru nedojde k napadení počítačů v lokální síti. [6]

Chráněná podsíť

Přestože je pojem chráněná podsíť často zaměňován s DMZ, nejedná se o totéž. Chráněné podsítě jsou speciální oddělené sítě, které jsou připojeny k rezervovanému rozhraní firewallu či jinému zařízení sloužícímu k filtrování. V této síti jsou obvykle umístěny servery, které jsou dostupné z Internetu a mají být odděleny od síťových zařízení sloužících jen pro potřeby vnitřních uživatelů dané organizace. Jejím účelem není oddělit lokální chráněnou síť od vnějšího přístupu, tak jak je tomu u DMZ, ale zdůraznit zabezpečení pro služby v ní umístěné.

V této zabezpečené síti jsou zásadně provozovány veřejné služby typu DNS, web, pošta atd. Servery tohoto typu musí být lépe zabezpečeny a zásadně se musí dodržovat doporučené postupy pro jejich ochranu, protože jsou cílem většiny útoků.

NAT

Překlad síťových adres neboli NAT (Network Address Translation) je způsob modifikace provozu sítě přes router, při kterém se přepíše zdrojová nebo cílová IP adresa (často se upraví i číslo TCP/UDP portu průchozích paketů) [11]. Prakticky to znamená, že adresy z vnitřní sítě jsou překládány na veřejnou IP adresu, přičemž si router překládanou adresu uloží do tabulky pod náhodným portem. Při odpovědi z vnější sítě do vnitřní se vyhledá v tabulce příslušný port a data se přepošlou na adresu vnitřní sítě přiřazenou k danému portu. Mechanismus NAT se dělí na dva typy – SNAT neboli Source NAT a DNAT neboli Destination NAT. O nich si však něco řekneme až v dalším v průběhu práce.

Zřejmou výhodou využití překladu adres je zvýšení bezpečnosti stanic připojených za NATem ve vnitřní síti – potenciální útočník totiž nezná opravdovou IP adresu. Další výhodou je umožnění připojení více stanic na jednu veřejnou IP, čímž se řeší nedostatek přidělených veřejných IP adres.

3.3 Firewally

Firewall je hardwarový a softwarový prostředek fyzicky i logicky oddělující důvěryhodnou síť od nezabezpečené nedůvěryhodné sítě. Uzly sítě neoddělené firewallem jsou vystaveny hrozbám, které s sebou přináší použití nezabezpečených protokolů a služeb Internetu. Implementace bezpečnostní politiky firemní sítě je záležitostí každého uzlu a pro dosažení požadované úrovně zabezpečení spolu musí všechny uzly kooperovat. [7]

Firewally umožňují část bezpečnostní politiky soustředěně implementovat do jednoho bodu, což je většinou i nejlevnější řešení. Usnadňují implementaci bezpečnostní politiky tam, kde lze protiopatření budovat na základě sledování a ovládání toku dat mezi chráněnou sítí a nedůvěryhodným okolím. O probíhajících útocích může firewall online informovat správce sítě a navíc pomocí zaznamenaných statistik lze analyzovat a vyhodnotit rizika. [7]

Firewall může zabránit v přístupu ke všem uzlům z vnějšku chráněné sítě, až na definované výjimky jako třeba poštovní servery. Také může zamezit tomu, aby se skrz zranitelné služby jako NFS (protokol pro vzdálený přístup k souborům přes počítačovou síť)

nebo NIS (protokol pro distribuci konfiguračních dat v síti) zpřístupňovala důvěryhodná síť zvenku. Slouží také jako jednoduché protiopatření proti útoku na soukromí – lze snadno zakázat používání služeb finger (např. útočníkovi může poskytnout informace o době posledního přihlášení uživatele) a DNS (znalost vazby jména uzlu s jeho síťovou adresou).

Z pohledu hardware je typický firewall buď počítač nebo směrovač alespoň se dvěmi síťovými kartami (nebo jiným zařízeními pro provoz TCP/IP, například sériovou linkou). V případě směrovače je softwarové vybavení nahráno přímo v ROM paměti směrovače, v případě počítače musí být zajištěno vhodným operačním systémem, nejčastěji na bázi Unix/Linux. Ve firewallu je implementována bezpečnostní politika, autentizační mechanismus, aplikační brána a filtr paketů. [7]

3.3.1 Negativa použití firewallů

Základní podmínkou je, aby firewall byl jediným přístupovým bodem do zabezpečované sítě. Proto se musí zajistit, že v chráněné síti nebudou používány modemy a žádné další pevné spojení sítě s Internetem. Pokud se někde v organizaci používá přímé spojení ven, pak nasazení oddělovacích uzlů postrádá smysl. Firewall je určen k ochraně před útoky zvenčí, což znamená, že lokální zdroje, které mají být přístupné jen z lokální sítě, zabezpečeny nejsou.

Firewall neřeší všechny problémy zabezpečení, je pouze jedním z mnoha prakticky použitelných bezpečnostních mechanizmů. Pomocí firewallu můžeme zablokovat použití služeb jako telnet, FTP, X Windows, NFS atd., ovšem následná nedostupnost těchto služeb není problém firewallů, ale je dána bezpečnostní politikou, kterou implementují.

Firewally nezabrání útokům vedeným přes data, jako např. prostřednictvím WWW služeb, pošty atd. Riziko hrozí i v případě přenosu videa a hlasových dat zapouzdřených do dále neanalyzovatelných paketů. Komprimované a zašifrované data mohou obsahovat viry.

Firewall sám musí být zabezpečený proti útoku z všech připojených sítí a musí být chráněn i proti přímému přístupu ke konzoli. Kompromitace firewallu může mít pro jinak slabě chráněnou síť katastrofální následky.

3.3.2 Paketové filtry

Paketový filtr je jednoduchý firewall, který je většinou implementován i ve směrovačích, přepínačích atd. Analyzuje IP hlavičku a podle adresy, portu odesílatele a příjemce ve svých tabulkách zjistí, jestli paket smí nebo nesmí být odeslán dál. Pakety, které splňují předem daná kritéria směruje dál a ty, které je nesplňují ničí nebo odmítá. Umožňuje tedy jednoduché

rozlišení mezi povolenou a nepovolenou komunikací. Dokáže rozpoznat jednotlivé počítače nebo sítě, ale neumožňuje autentizaci uživatelů. Paketový filtr je zařízení operující na síťové úrovni. [7]

3.3.3 Aplikační brány

Aplikační brána pracuje na aplikační úrovni – paketovou vrstvu nekontroluje. Zabývá se obsahem IP paketů, ne IP pakety samotnými. Aplikačních bran se může provozovat v několik současně, každá může řídit jinou službu. Typické nasazení aplikačních bran je v oblasti e-mailu a WWW.

Aplikační brána funguje na jiném principu než paketový filtr (viz. kapitola 3.3.3) – zadrží všechny pakety a pro povolená spojení sama znovu naváže spojení a případně modifikuje tok dat. Velkou výhodou aplikačních bran proti filtrování paketů je podstatně vyšší bezpečnost, protože přístup k důležitým službám se umožňuje až na základě autentizace uživatele a ne na základě poznání IP adresy počítače. [7]

4 Síťové topologie

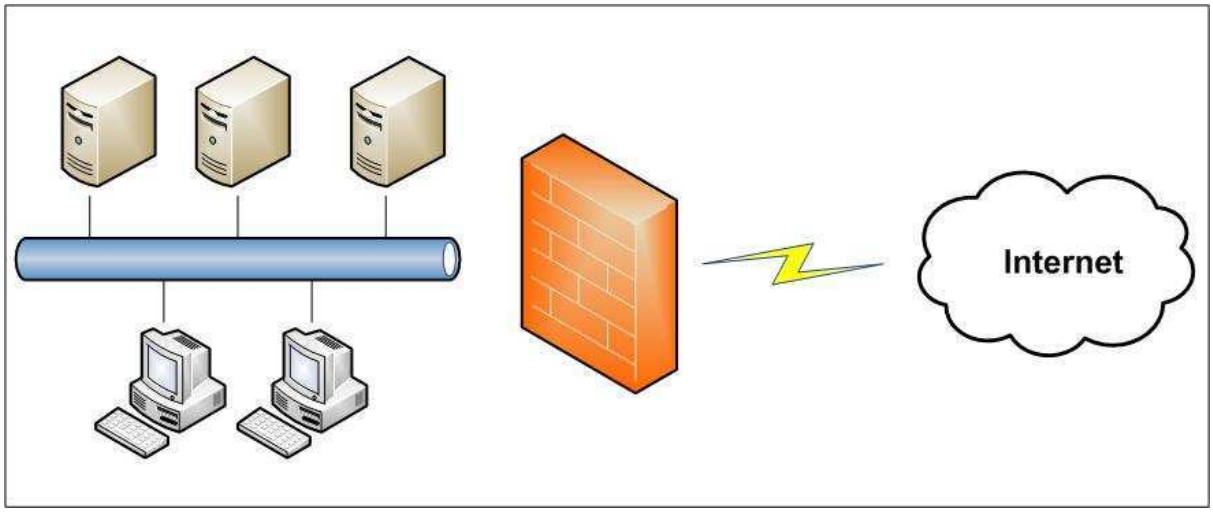
Topologie sítě je dána zvolenou architekturou sítě, pod kterou spadá také rozložení jednotlivých stanic, serverů i bezpečnostních prvků. Architektura se odvíjí především od rozsahu sítě. V této práci je zadán návrh sítě pro středně velkou infrastrukturu, proto budeme brát v úvahu klasické schéma sítě s typickými prvky v podobě pracovních stanic, serverů poskytujících služby zaměstnancům i klientům v Internetu a jedním internetovým připojením.

Síť chceme mít kvalitně zabezpečenou a proto musíme při jejím návrhu postupovat tak, aby použité prvky zabezpečení byly efektivně využity. Základem návrhu je řešení otázky, kam umístit firewall, který je pro zabezpečení sítě nezbytný. Chceme, aby firewall střežil a zároveň i tvořil hranici mezi podnikovou sítí a Internetem. Od této funkce se odvíjí jeho název – hraniční firewall.

Pro středně velkou síť je vhodné použití jen jednoho firewallu. Více firewallů se používá v rozsáhlých sítích, kde např. chceme mít odděleny jednotlivé části společnosti nebo chceme oddělit používání velmi citlivých služeb. Pro použití jednoho firewallu se nám obecně nabízí tři topologie založené na jeho umístění.

4.1 Firewall před službami

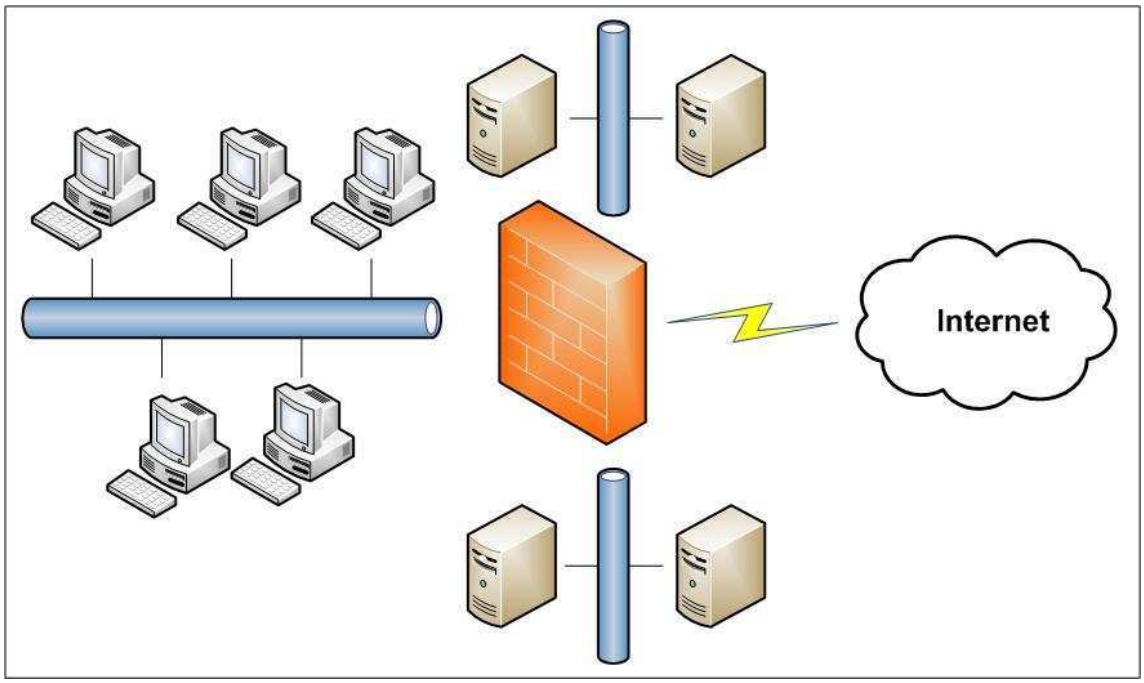
V tomto provedení je ochráněna celá vnitřní síť včetně serverů. Firewall ale blokuje i veřejně dostupné služby serverů jako web, mail a DNS, protože zabraňuje v přístupu všem neznámým připojením z Internetu. Abychom mohli tyto služby zprovoznit, musíme povolit přesměrování portů náležících těmto službám. To s sebou přináší ale bezpečnostní riziko, protože otevřené porty mohou být zneužity k útoku na síť a přes ně by mohl potencionální útočník proniknout do naší vnitřní sítě. Stále je ale toto řešení výrazně bezpečnější než možnost v kapitole 4.2, kde je firewall umístěn za službami. Schéma topologie s firewallu před službami je na obrázku číslo 7.



Obr. č. 7 – Firewall před službami

4.2 Firewall za službami

Jak vidíme na Obr. č. 8, servery poskytující služby jsou vloženy před firewall a tím pádem nemají ochranu. Proto u nich musíme dbát na důkladnější softwarové zabezpečení a počítat se zvýšeným rizikem a vystavením útokům. Musíme také zvážit uložení důležitých nebo citlivých informací na tyto servery, protože si nemůžeme být jistí jejich stoprocentí ochranou. Výhodou tohoto řešení je lepší ochrana vnitřní sítě.



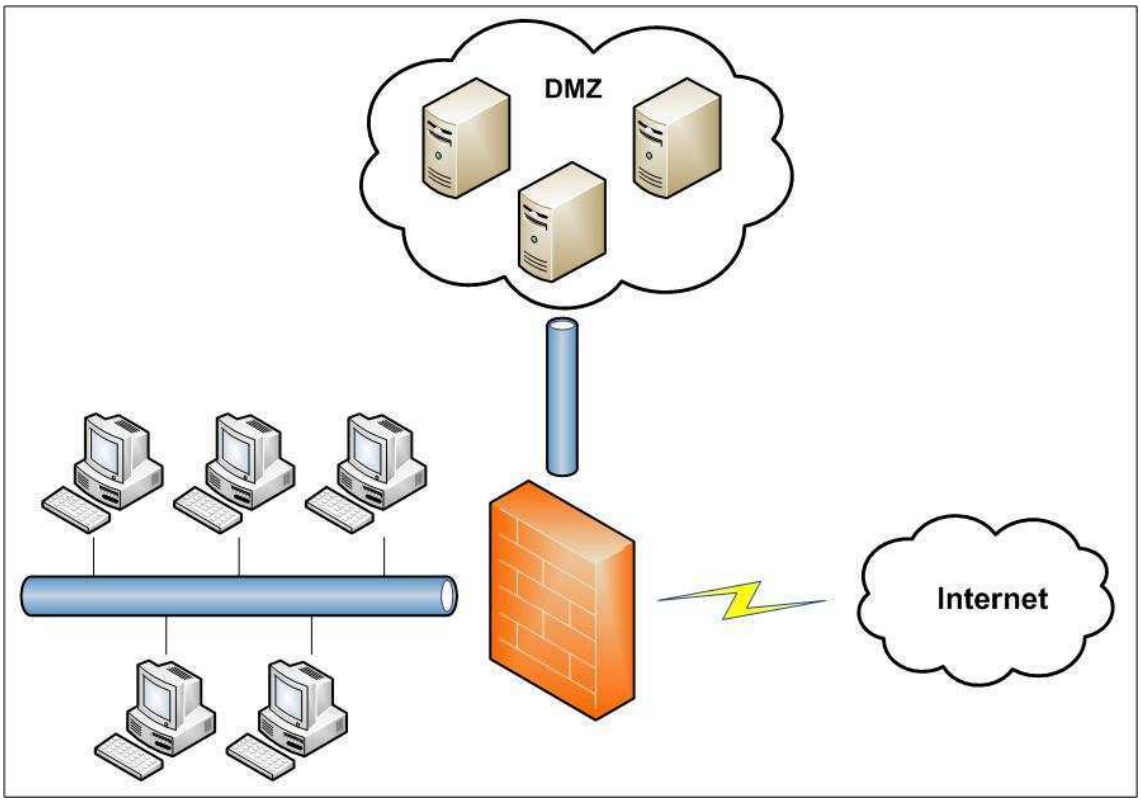
Obr. č. 8 – Firewall za službami

4.3 Firewall s DMZ

Toto řešení je v praxi často používané, protože je výhodným kompromisem mezi zabezpečením sítě a její složitostí. Zvolíme-li tuto topologii, tak se vyhneme nevýhodám a úskalím, které přináší předešlé dvě řešení.

Firewall podporující DMZ (viz. kap. 3.2) má několik rozhraní (alespoň tři), které náleží jednotlivým sítím. Pro vnitřní klienty je vyhrazeno interní rozhraní, DMZ rozhraní je přiřazeno interní síti s veřejnými servery a vnější rozhraní náleží Internetu. Mezi těmito sítěmi musí být aktivováno filtrování a nesmí docházet ke směrování. Firewall ochraňuje DMZ tak, že filtruje všechen nelegitimní provoz a propustí do DMZ jen ty dotazy, které mají příslušné oprávnění. DMZ je od vnitřní sítě oddělena úplně nebo alespoň částečně, což zaručuje ochranu vnitřní sítě v případě úspěšného útoku na veřejné servery. Důležité je komunikaci mezi jednotlivými částmi sítě a DMZ minimalizovat. Vnější rozhraní by mělo mít přístup do DMZ jen přes konkrétní porty a vnitřní rozhraní by mělo mít přístup jen pokud je to třeba.

Tuto topologii jsem si vybral pro návrh sítě, protože je ideální pro středně velkou infrastrukturu. Je dostatečně bezpečná, její realizace není komplikovaná ani nepřiměřeně finančně náročná a je velmi často používána v praxi pro tento typ středně velké sítě.



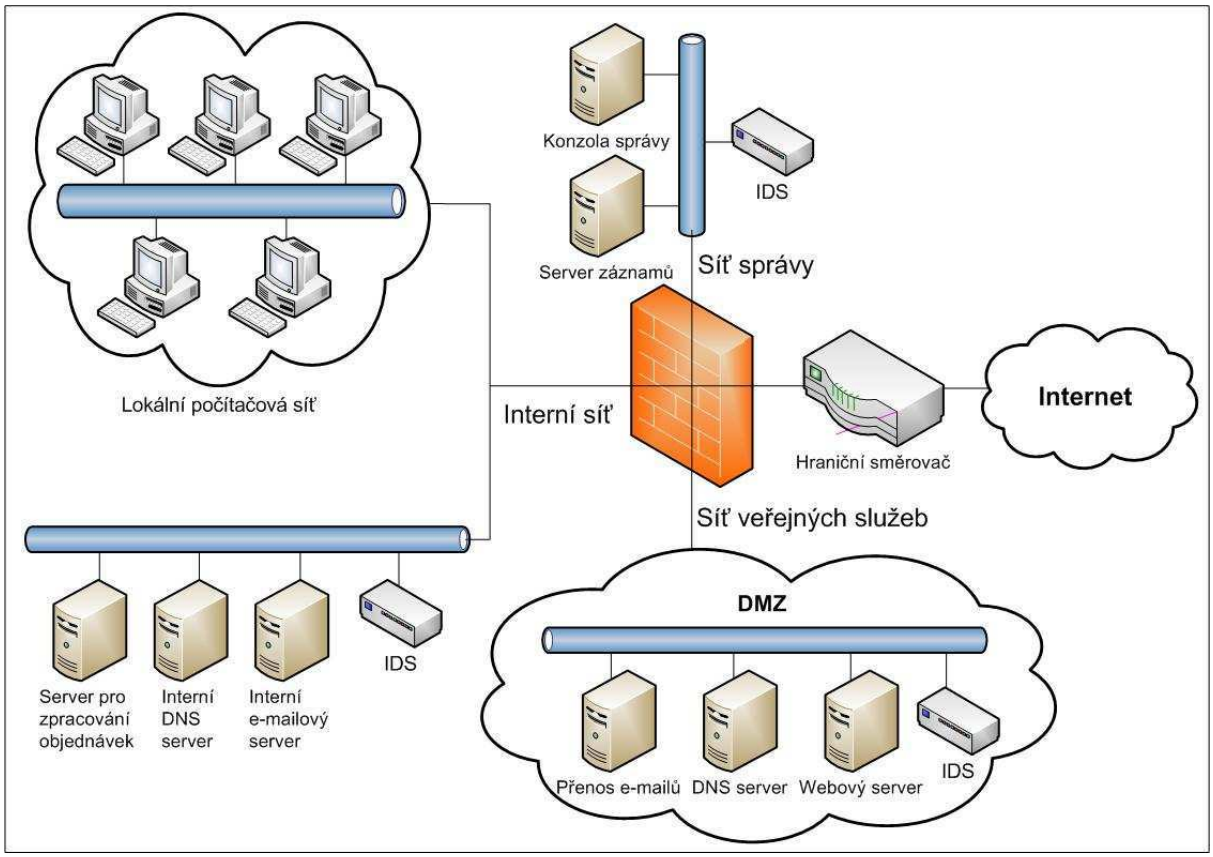
Obr. č. 9 – Firewall s DMZ

5 Návrh sítě

Navrhovaná síť je určená pro organizaci střední velikosti, která chce důsledné zabezpečení sítě, jejich prvků a také citlivých dat o zákaznících. Tato organizace má svou prezentaci na Internetu a své výrobky přes Internet i prodává. Proto se bojí útoků, které by mohly ohrozit data jejich zákazníků. Jejich stránky denně navštíví několik set návštěvníků, takže jde o středně velký provoz. Zaměstnanci této organizace využívají e-mailových služeb a mají možnost přístupu na externí internetové stránky. Organizace klade velký důraz na zabezpečení a podle toho vyčlenila na realizaci návrhu přiměřeně velký rozpočet.

Návrh sítě je na obrázku č. 10. Některé rysy a výsledky návrhu nejsou z obrázku patrné, proto se jim podrobněji věnuji v dalším textu. Pro lepší přehlednost byly jednotlivé služby umístěny na samostatné stanice. V reálném nasazení by bylo možné počet stanic výrazně snížit umístěním více služeb na jednu stanici/server.

Také musím zdůraznit, že se jedná o jedno z mnoha řešení, jak lze síť navrhnout. V mé další práci, kde budu návrh sítě realizovat, se zaměřím především jen realizací hraničního prvku sítě.



Obr. č. 10 – Návrh sítě

Popis hlavních rysů návrhu

- Hraniční směrovač poskytuje základní filtraci propojení sítě a Internetu. Tvoří první linii kontroly přístupu do firemní sítě.
- Za hraničním směrovačem následuje proxy firewall s podporou DMZ, který obstarává převážnou část řízení přístupu k síti. Jedná se o hlavní zabezpečovací prvek sítě.
- Firewall rozděluje síť na jednotlivé segmenty. Síť je rozdělena na segment pro správní systémy, segment veřejných služeb a segment privátních služeb.
- Správní síť je tvořena servery pro správní konzolu a záznamy.
- Interní síť se skládá z pracovních stanic a privátních serverů, ke kterým nemají externí uživatelé přístup.
- Síť veřejných služeb obsahuje servery poskytující služby externím uživatelům (uživatelé z Internetu).
- Jednotlivé segmenty sítě jsou zabezpečeny systémem IDS (viz. kap. 3.2), kterému náleží sledování a hlášení podezřelých a nepovolených aktivit v síti.

- Pokud je zjištěna nepovolená nebo neobvyklá činnost v síti, je zaslána serveru záznamů, který vygeneruje výstrahu.
- Všechna bezpečnostní zařízení sítě jsou konfigurována přes správní konzolu.
- DNS server je rozdělen na dvě části. Interní DNS server obsahuje záznamy pro všechny systémy, ale externí uživatelé k němu nemají přístup. Pro rozeznávání jmen veřejných služeb je vyhrazen veřejný DNS server.
- Webový server ukládá data o zákaznících a jejich transakcích jen na omezenou dobu. Tato data jsou předána ke zpracování a vyhodnocení serveru pro zpracování objednávek. Jakmile tento server dostane data ke zpracování, jsou vymazána z webového serveru.
- Server pro přenos e-mailu uchovává přijaté e-maily ve své paměti, dokud nejsou převzaty interním e-mailovým serverem.
- Vnější okolí sítě je považováno za zónu nedůvěryhodnou a potenciálně nebezpečnou, proto se vnitřní systémy sítě nemohou připojit do této zóny. Jedinou výjimkou přístupu do dané oblasti je připojení pracovních stanic z lokální počítačové sítě. Pracovním stanicím je umožněn přístup na externí webové a FTP servery a spojení je realizováno přes připojení k proxy firewallu.

6 Realizace

6.1 Úvod

Cílem této kapitoly je popsat a vysvětlit jednotlivé kroky realizace a nastavení síťového hraničního směrovače pomocí operačního systému Debian GNU/Linux. Tuto distribuci GNU/Linuxu jsem si vybral pro jeho stabilitu, snadnou údržbu a časté využití pro serverové instalace. Přestože je Debian známý pro svou konzervativnost, jedná se o jednu z nejrozšířenějších distribucí na světě. Použil jsem poslední stabilní verzi Debian 5.0 a to jen v nejzákladnější instalaci bez desktopového prostředí. Pracoval jsem tedy jen přes terminál.

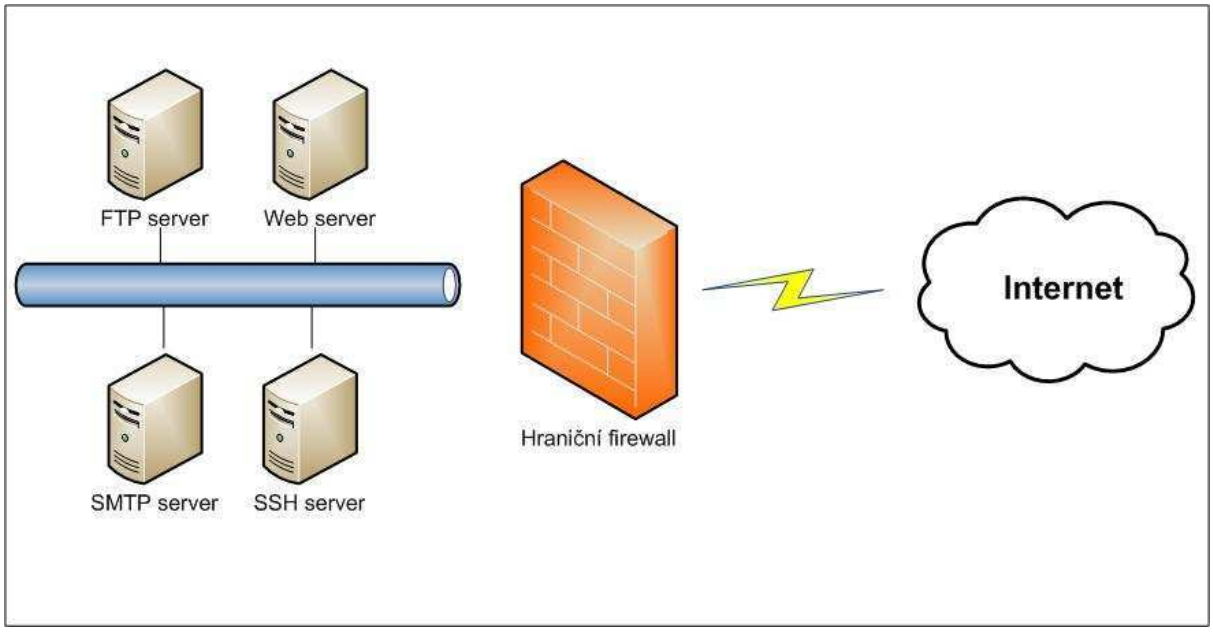
Rozhodl jsem se použít virtuální stroj místo fyzického, kvůli řadě výhod, které technologie virtualizace přináší. Jde především o větší přehlednost, efektivnější ovládání a možnost pracovat s více běžícími operačními systémy zaráz. Virtualizace umožňuje na jednom fyzickém počítači provozovat více operačních systémů odlišných platforem včetně jejich vlastních aplikací. Pro virtualizaci jsem použil bezplatný virtualizační nástroj VirtualBox

(verze 2.1.4), určený pro kompletní simulaci hardwarového prostředí. Některé prvky jsem testoval i ve virtualizačním nástroji VMware Workstation 6.5.2.

6.2 Model realizovaného návrhu

Základem realizovaného návrhu je hraniční směrovač, plnící zároveň i funkci firewallu/paketového filtru. Odděluje vnější prostředí Internetu od vnitřní lokální sítě, ve které jsou umístěny služby typu FTP, SSH, SMTP a Web server. Adresy stanic lokální sítě jsou jsou překládány pomocí NAT na jedinečnou veřejnou IP adresu, která slouží pro vstup do Internetu.

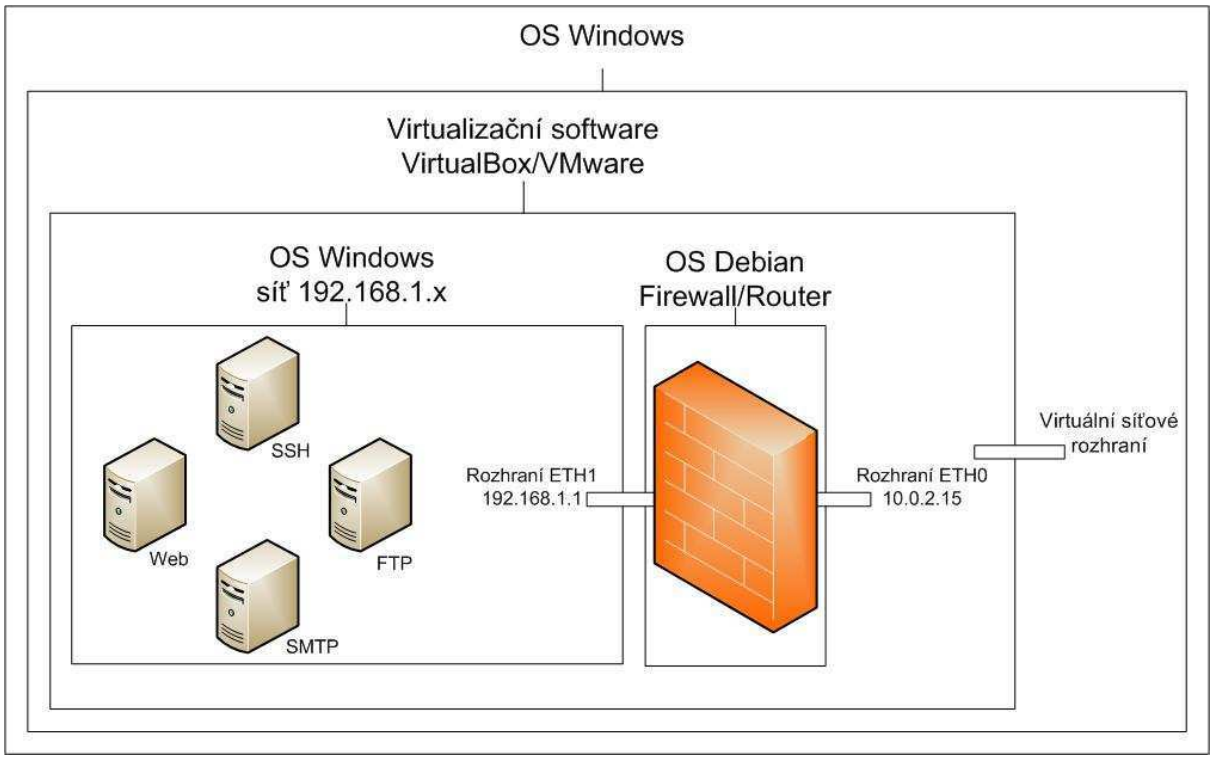
Na obrázku č. 11 na další straně je zjednodušený model návrhu:



Obr. č. 11 – Zjednodušený model návrhu

Z obrázku č. 12 je patrné, jak jsem měl model rozvržen při realizaci. Virtuální OS Windows představuje vnitřní síť, ve které jsou umístěny jednotlivé služby. Tomuto OS byla přes virtualizační software přiřazena jedna síťová karta a to pro komunikaci s vnitřní sítí LAN. OS Debian je rovněž připojen jedním síťovým rozhraním k síti LAN a druhým rozhraním k NAT. Debian má tedy přímé připojení k Internetu a to přes překlad adres NAT. Windows se k Internetu připojuje přes Debian, který představuje hraniční prvek sítě řídící komunikaci mezi vnitřním a vnějším síťovým prostředím. Aby spojení vnější a vnitřní sítě bylo možné, musela se provést řada kroků, které si postupně popíšeme.

Aby bylo možné připojení virtuálních strojů k Internetu, virtualizační software vytváří v operačním systému síťová připojení, která fyzicky neexistují – jsou virtuální. Slouží k přemostění spojení mezi virtuálními stroji a Internetem.



Obr. č. 12 – Skutečný model návrhu

6.3 Iptables

Hlavním nástrojem, který jsem při realizaci a konfiguraci firewallu/směrovače používal, je administrátorský nástroj iptables, který v systémech Linux a Unix umožňuje plně pracovat se síťovou komunikací, vytvořit a nastavit různé druhy firewallů (stavový, transparentní,…) nebo sdílet připojení k Internetu.

Iptables obecně slouží k administraci pravidel filtrovaní v jádru Linuxu. Předdefinované řetězce pravidel jsou zapsány v tabulkách a my můžeme do těchto tabulek přidávat a definovat naše vlastní řetězce. Řetězec je označení pojmenovaného seznamu pravidel. Každým pravidlem jsou stanoveny podmínky, kterým musí daný paket vyhovovat. Zároveň je v pravidlech označen cíl nebo cílová akce, která bude vykonána v případě, že paket vyhovuje podmínkám. Jako cíl můžeme například stanovit ACCEPT nebo-li akceptování paketu či DROP zahození paketu. Pokud paket nevyhovuje podmínkám pravidla, je „předán“ dalším pravidlům v daném řetězci a paket takto postupně prochází celým řetězcem dokud některému

pravidlu nevyhoví. Pokud paket není propuštěn žádným z pravidel, vrací se do nadřazeného řetězce, odkud byl daný řetězec volán. Jestliže se takto paket dostane na konec definovaného řetězce, je mu přiřazen implicitní cíl příslušního řetězce – u firewallů standartně DROP. [9] Program iptables voláme vždy s několika parametry. Základní syntaxe zápisu je [9]:

```
iptables [tabulka] [akce] [chain] [ip_část] [match] [target] [target_info]
```

Funkce a význam jednotlivých použitých parametrů iptables budu vysvětlovat zároveň s postupem, věřím, že to tak bude nejsrozumitelnější. Parametry jsem po jednom nezadával do konzole, ale postupně jsem psal skript, který obsahoval a prováděl všechny potřebné úkony a parametry.

6.4 Základní úkony

Předtím, než jsem se začal zabývat samotnou konfigurací firewallu a směrovače, bylo nutné vykonat několik úkonů, aby realizace a správná funkčnost firewallu, byla vůbec možná. Jako první bylo třeba nakonfigurovat síťová rozhraní v OS Debian i v OS Windows XP, nastavit přeposílání paketů mezi sítěmi a správně přiřadit skriptu přístupová práva.

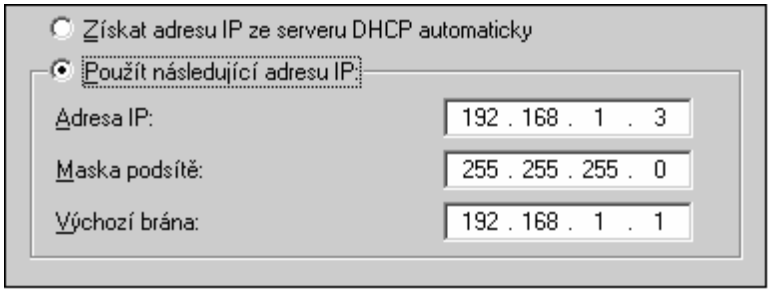
6.4.1 Nastavení síťových rozhraní

V OS Debian jsou informace a nastavení síťových rozhraní uloženy v souboru /etc/network/interfaces, který slouží k automatizovanému ovládání síťových rozhraní počítače. [10]

V základním nastavení je nakonfigurována místní smyčka lo (The Loopback Network Interface) a primární síťové rozhraní eth0 (The primary network interface). Rozhraní eth0 budeme využívat pro komunikaci s vnějším prostředím Internetu. Toho rozhraní není třeba konfigurovat, standartně je nastaveno na přiřazení IP adresy serverem DHCP. Je však nutné vytvořit nové rozhraní eth1, které bude sloužit k připojení vnitřní privátní sítě. U rozhraní eth1 stačí přiřadit statickou IP adresu a masku sítě. Zápis nového síťového rozhraní je na další straně a vypadá následovně:

```
auto eth1
iface eth1 inet static
address 192.168.1.1
netmask 255.255.255.0
```

V OS Windows jsem síťové rozhraní konfiguroval přes Síťová připojení -> Připojení k místní síti -> Vlastnosti -> Protokol sítě Internet (TCP/IP). Zde je standartně nastaveno získání IP adresy z DHCP serveru, my toto nastavení však musíme změnit na manuální nastavení, čili možnost Použít následující adresu IP. Rozsah adres vnitřní sítě nám umožňuje připojit až 254 stanic, čili IP adresu zařízení můžeme zvolit vpodstatě jakoukoliv z intervalu 192.168.1.2 až 192.168.1.255. Výchozí bránu tvoří síťové rozhraní eth1, které jsme nastavovali v předešlém kroku a proto zapisujeme jeho síťovou adresu. Celkové nastavení vypadá následovně:



Obr. č. 12 – Nastavení protokolu TCP/IP

6.4.2 Nastavení přeposílání

Forwarding neboli přeposílání umožňuje, aby všechny pakety na jednom síťovém rozhraní (například eth0) mohli být přeposlány na jiné síťové rozhraní (např. eth1). To nám umožňuje využít počítač s Linuxem k propojování sítí nebo ke směrování síťového provozu.

Z bezpečnostních důvodů je ovšem možnost přeposílání paketů mezi sítěmi standartně vypnutá. Pokud chceme přeposílání povolit, nejjednoduší možností je zadat do konzole parametr:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Tento parametr nahradí v souboru ip_forward hodnotu 0 (která značí vypnutí přeposílání) za hodnotu 1, která povoluje přeposílání paketů. Tato hodnota však bude při dalším spuštění systému vrácena zpět na hodnotu 0. Proto jsem parametr, který zpřístupňuje přeposílání, musel zařadit na začátek skriptu, který se bude spouštět automaticky při každém bootu systému.

6.4.3 Skript

Skript je linuxový ekvivalent pro dávkový soubor (OS Windows) a jeho tvorbu lze shrnout do tří kroků – návrh skriptu, zapsání do textového souboru a nastavení určitého atributu souboru, který určí, že se jedná o spustitelný soubor – skript. Pro zapisování jsem používal editor souborů obsažený v souborovém manažeru Midnight Commander.

Skript se při vytváření zapisuje jako běžný textový soubor, aby však bylo možné ho spustit, je třeba nastavit správně jeho atributy. Atributy se mění přes příkaz chmod, který obecně slouží ke změně přístupových práv souboru. Uživatelské skupiny jsou rozděleny z hlediska přístupu do tří skupin: **owner, group, other** a každá z těchto skupin může mít tři odlišná povolení **read** (čtení), **write** (zápis) a **execute** (spuštění souboru).

Tyto nastavení jsou pak chmod vyjadřovány následovně:

400 - owner má read povolení
200 - owner má write povolení
100 - owner má execute povolení

040 - group má read povolení
020 - group má write povolení
010 - group má execute povolení

004 - other má read povolení
002 - other má write povolení
001 - other má execute povolení

Sečtením všech parametrů dohromady se získá požadovaná hodnota chmod. Po skriptovém souboru požadujeme, aby byl owner měl povolení read, write a execute zatímco group a other mají povolení jen execute a read, což vyjádříme jako součet:

400+200+100+40+10+4+1 = 755

Čili pro soubor skript.sh do konzole zapíšeme:

```
chmod 755 skript.sh
```

6.5 Ping

Pro ověření, že jsou obě virtuální stanice správně nastaveny, je nejlepší použít program PING (Packet InterNet Groper). Slouží k ověření funkčnosti spojení mezi dvěma síťovými rozhraními v počítačové síti fungující na rodině protokolů TCP/IP. Ověření probíhá pomocí periodického zasílání IP datagramů a čeká na odezvu protější strany. Při úspěšném navázání spojení a obdržení odpovědi se vypíše latence – délka zpoždění a statistický souhrn.

Pro zajištění správné funkčnosti programu ping, stačí sestavit relativně jednoduchý skript. Na jeho začátku si pro jednodušší zapisování definujeme cestu k iptables a to přes zápis:

```
IPTABLES=/sbin/iptables
```

Je nutné taky povolit přeposílání paketů mezi sítěmi, zápis tohoto kroku jsme si uvedli v kapitole 7.4.2. V následujícím kroku je vhodné vymazat všechny předešlá uživatelsky definovaná pravidla iptables, které by mohli mít negativní nebo nežádoucí vliv. Použijeme příkazy:

```
$IPTABLES -F
$IPTABLES -X
```

Dalším nastavením, kterým je nutné se zabývat, jsou tabulky. Existují tři základní – **filter**, **nat** a **mangle**. Automaticky se používá filter, pokud to přímo nedefinujeme jinak. V této tabulce jsou řetězce INPUT, OUTPUT a FORWARD. Jak už napovídá název, pro pakety směřující dovnitř sítě, se uplatňuje řetězec INPUT. Pro odchozí pakety se uplatňuje OUTPUT a FORWARD se používá tehdy, když stanice slouží jako router a přeposílá pakety mezi sítěmi.[9] Musíme také brát v potaz, že to co prochází přes FORWARD, není ovlivněno řetězci INPUT ani OUTPUT.

Pro firewall je nejvhodnější uplatnit koncepci ,,co není výslovně povoleno, to je zakázáno‘‘.[11] To prakticky znamená, že by implicitním nastavením všech řetězců mělo být zahazování (DROP) všech paketů nevyhovujících stanoveným pravidlům. Šlo by to i řešit opačně a to povolit všechno a filtrovat jen nežádoucí toky, ale tento způsob by při větším provozu mohl být značně komplikovaný a i méně bezpečný.

Výchozí politika řetězce se pro iptables zapisuje pomocí přepínače –p a volby jsou buď ACCEPT (povolit) nebo DROP (zahodit). Příkazy by tedy měly vypadat:

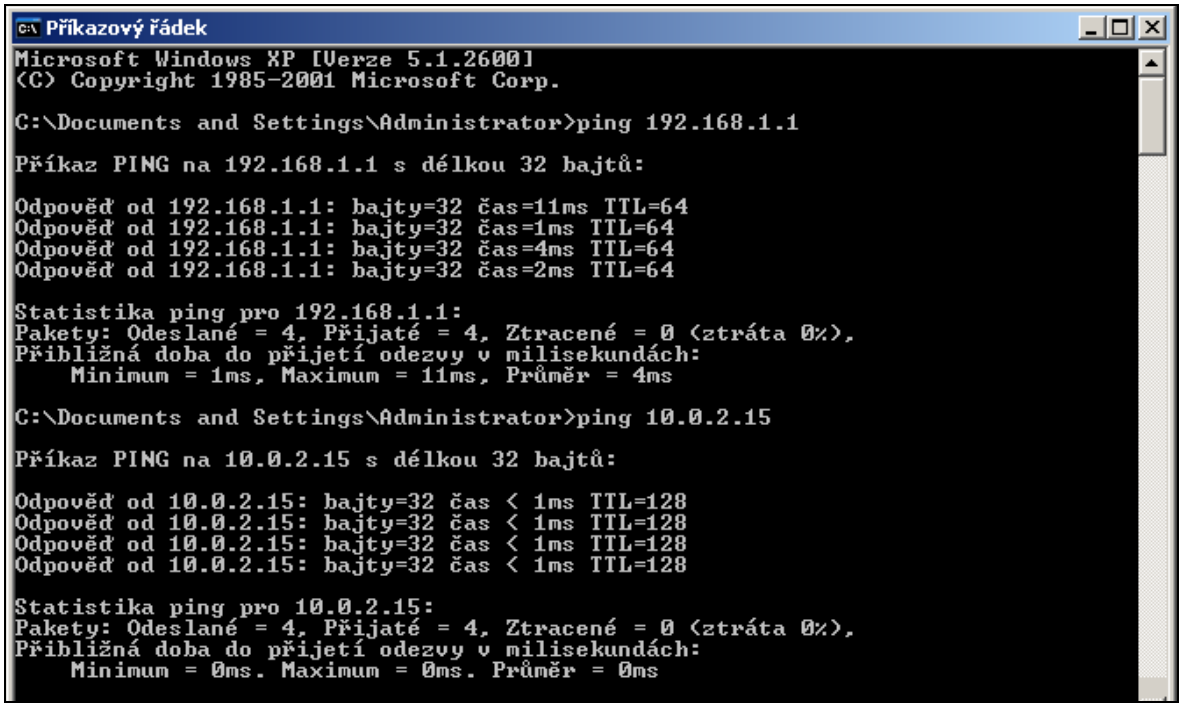
```
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP
```

Abychom si nastavování pravidel ulehčili, můžeme řetězci OUTPUT přiřadit ACCEPT, protože není nutné filtrovat odchozí provoz ze sítě. Síť chceme ochránit před nežádoucími spojeními směřujícími z Internetu dovnitř sítě.

Jelikož chceme programem ping ověřit komunikaci mezi vnitřní sítí a firewallem, můžeme povolit věškerou komunikaci posílanou přes rozhraní eth1, které propojuje důvěryhodnou vnitřní síť a firewall. Pro toto nastavení využijeme přepínače –A, který značí přidání nového pravidla na konec řetězce a přepínače –i označujícího vstupní zařízení. Zadáme tedy příkaz:

```
$IPTABLES -A INPUT -i eth1 -j ACCEPT
```

Tímto máme vyřešené základní úkony nastavení pro ověření funkční komunikace mezi síťovými rozhraními. Na virtuálním stroji představujícím vnitřní síť stačí spustit příkazový řádek a zadat ping a adresu cílové stanice. Nejdříve zadáme adresu výchozí brány eth1, abychom ověřili, že komunikace na této úrovni je správná a po správné odezvě prověříme komunikaci s rozhraním eth0:



Obr. č. 13 – Ověření odezvy síťových rozhraní

6.6 Nastavení firewallu

Nyní máme základ skriptu, na který budeme navazovat a také jsme ověřili, že sítě mezi sebou bezchybně komunikují. Máme tedy nastavenou směrovací tabulku včetně řetězců a komunikaci mezi vnitřní sítí a firewallem. Nyní přidáme pravidlo pro přeposílání paketů mezi rozhraním vnitřní sítě a rozhraním vnější sítě:

```
$IPTABLES -A FORWARD -i eth1 -o eth0 -j ACCEPT
```

Dále povolíme pakety příchozích spojení z vnější sítě pro spojení, která již byla navázána. K tomu slouží stavové parametry RELATED a ESTABLISHED. RELATED umožňuje již existující konexi využít i jiné porty než byly pro spojení původně určeny, přestože tyto porty nemusí být otevřeny (zejména vhodné pro případ využití FTP). ESTABLISHED umožňuje komunikaci s již navázaným spojení tak, jak si to řídí obě strany. Zápis příkazu je:

```
$IPTABLES -A FORWARD -i eth0 -o eth1 -m state –state \
ESTABLISHED,RELATED -j ACCEPT
```

Pro případ příchozích paketů, které nemohou být identifikovány, jsou vadné, poškozené nebo nenáleží žádnému známému spojení, se používá označení stavu INVALID. Tyto pakety rozhodně nemáme zájem propouštět do sítě, proto je budeme zahazovat:

```
$IPTABLES -A FORWARD -i eth0 -o eth1 -m state --state INVALID\
-j DROP
```

6.6.1 Ochrana proti hackerským technikám

Nyní máme základy firewallu vyřešeny a můžeme přidat ochranu proti jednoduchým, avšak často používaným hackerským technikám, které nezřídka umožní útočníkům proniknout do chráněné sítě.

První takovou technikou je IP spoofing nebo-li falšování zdrojové IP adresy, kdy útočník maskuje svou pravou IP adresu nebo předstírá, že je někdo jiný. [11] Původce podvržených paketů jistě nehodlá do naší sítě vstoupit s čistými úmysly, proto je vhodné všechny tyto pakety filtrovat na vstupu firewallu. Základní ochranu proti tomuto druhu útoku představuje

rp_filter, který blokuje pakety se zdrojovou adresou, která by dle směrovacích tabulek měla přijít z jiného dostupného síťového rozhraní. Aktivujeme jej příkazem:

```
echo 1 > /proc/sys/net/ipv4/conf/eth0/rp_filter
```

Další ochranu, kterou není složité realizovat, je obrana před hackerskou technikou SYN flooding. Jedná se o útok typu DoS (Denial of Service), který v tomto případě zahlcuje oběť velkým množstvím TCP segmentů s flagem typu SYN (požadavek na vytvoření nového spojení) [11]. Dříve nebo později oběť útoku podlehne a dochází k DoS. Tomu však můžeme předejít vytvořením pravidla, které omezuje počet nově vytvořených spojení za určitý časový interval. Pravidlo zapíšeme tak, aby pakety typu SYN byly propuštěny, pokud jich nebude více než 5 za sekundu a při překročení tohoto limitu budou filtrovány. Zápis nového řetězce syn_flood vypadá následovně [11]:

```
$IPTABLES -N syn_flood
$IPTABLES -A INPUT -i eth0 -p tcp --syn -j syn_flood
$IPTABLES -A syn_flood -m limit --limit 1/s --limit-burst 5\
-j RETURN
$IPTABLES -A syn_flood -j DROP
```

Podobnou technikou DoS je postup útok nazývaný Ping of death [11]. Je založena na zahlcení oběti velkým množstvím žádostí o odezvu pomocí programu Ping (viz. Kapitola 7.5). Každé zařízení má své omezené prostředky a tak by se i tímto postupem mohlo stát, že se zařízení zahlcené požadavky typu echo-request vypoví službu. Můžeme tomu však předejít, podobně jako v předešlém případě, stanovením limitu pro tyto žádosti v určitém časovém intervalu. Jedná se o omezení ICMP echo-requestů na maximální počet 5 za dobu jedné vteřiny. Pravidlo zapíšeme takto [11]:

```
$IPTABLES -A INPUT -p icmp --icmp-type echo-request -m limit \
--limit 1/s --limit-burst 5 -j ACCEPT
```

6.7 Nastavení směrování

V kapitole 7.5 jsme si zmiňovaly tři základní řetězce INPUT, OUTPUT a FORWARD. Kromě nich ovšem existují i další dva to PREROUTING a POSTROUTING. PREROUTING, jak napovídá název, je určený pro zpracování paketů v době před směrováním – čili zpracovává pakety určené jak pro přesměrování tak i pro lokální stroj. POSTROUTING naopak zpracovává pakety, které směřují odcházejí z našeho stroje. Ani jeden z těchto dvou zmíněných řetězců neslouží k filtrování paketů. Jsou využity při překladu adres NAT, o kterém jsme mluvili v kapitole 3.2. Co jsme si ale podrobněji nerozebrali, je rozdělení NAT na SNAT (zdrojový NAT) a DNAT (cílový NAT).

6.7.1 SNAT

SNAT využijeme z toho důvodu, že ve vnitřní síti máme umístěny stanice, které mají privátní typ adres, což není vhodně pro komunikaci s vnějším světem. Pro tuto komunikaci využijeme veřejnou IP adresu na síťovém rozhraní eth0. Musíme zajistit, aby se veškerá odchozí komunikace z naší sítě směrem ven tvářila, jako by pocházela se zařízení s veřejnou IP adresou. K tomu nám poslouží příkaz, který zajistí, že router u odchozích paketů zamaskuje (od toho odvozen název maškaráda – MASQUERADE) jejich původní IP adresu. Pakety potom budou po Internetu putovat s naší veřejnou IP adresou. Zápis je:

```
$IPTABLES -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

6.7.2 DNAT

DNAT se využívá tehdy když je třeba změnit IP adresu zařízení, pro které je je datagram určen. Jelikož v síti máme umístěny služby WWW, SMTP, SSH a FTP, musíme datagramy určené pro konkrétní služby třídit a zajistit, že se dostanou ke správnému adresátovi. Například neznámý uživatel z venkovní sítě Internetu, který se chce připojit na náš FTP server, se připojuje na naši veřejnou IP adresu. Jakmile se na náš router dostane požadavek o FTP spojení, router ho musí přesměrovat na odpovídající službu na příslušné síťové adrese a portu. Musíme tedy zařídit, aby router přepisoval cílové IP adresy datagramů jednotlivých služeb. O jakou službu se jedná router rozliší podle portu, pro který je segment určený.

Vnitřní síť máme navrženou tak, že jednotlivé služby běží na samostaných strojích, proto se u zápisu pravidel mění nejenom port, ale i příslušná IP adresa:

```
$IPTABLES -t nat -A PREROUTING -p tcp --dport 80 -i eth0 –j\
DNAT --to 192.168.0.2:80
$IPTABLES -t nat -A PREROUTING -p tcp --dport 21 -i eth0 –j\
DNAT --to 192.168.0.3:21
$IPTABLES -t nat -A PREROUTING -p tcp --dport 22 -i eth0 –j\
DNAT --to 192.168.0.4:22
$IPTABLES -t nat -A PREROUTING -p tcp --dport 25 -i eth0 –j\
DNAT --to 192.168.0.5:25
```

Musíme také však brát v úvahu, že když nějaký paket vyhoví pravidlu v tabulce NAT, neznamená to, že ho automaticky propustí paketový filtr. Jelikož máme řetězec FORWARD nastaven na zahazování paketů DROP, musíme zde povolit spojení pro služby v naší síti[11]:

```
$IPTABLES -A FORWARD -i eth0 -p tcp -d 192.168.0.2 --dport 80\
-m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -i eth0 -p tcp -d 192.168.0.3 --dport 21\
-m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -i eth0 -p tcp -d 192.168.0.4 --dport 22\
-m state --state NEW -j ACCEPT
$IPTABLES -A FORWARD -i eth0 -p tcp -d 192.168.0.5 --dport 25\
-m state --state NEW -j ACCEPT
```

6.8 Ověření nastavení

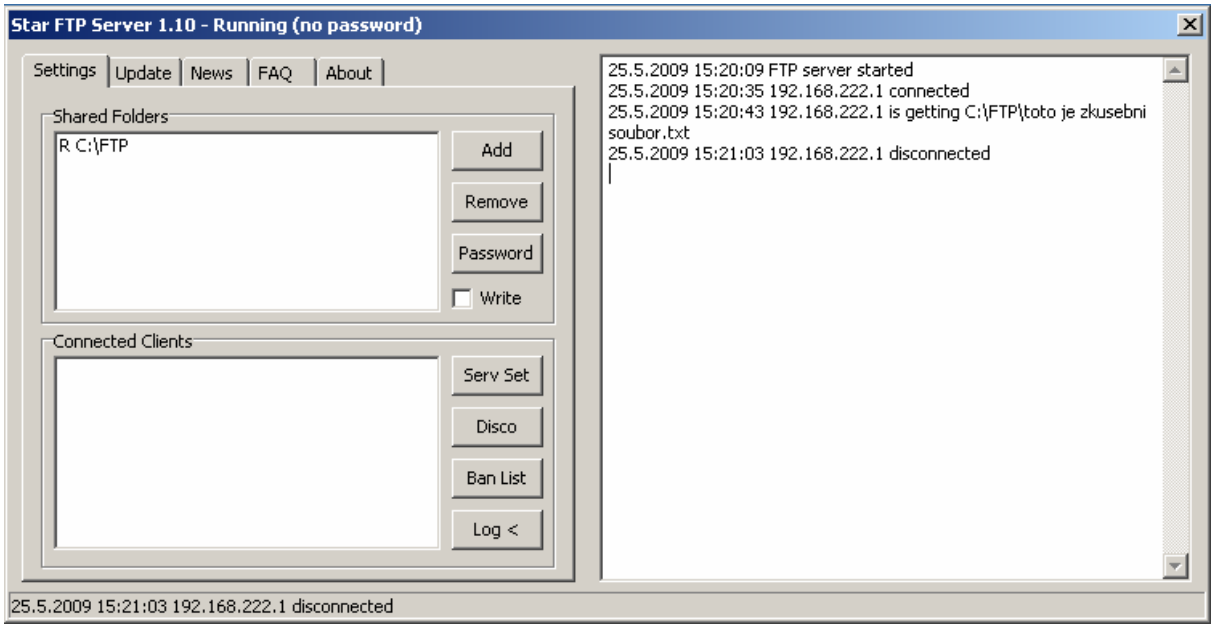
Potom, co jsme sepsali skript pro provoz a zabezpečení sítě, je nutné ho otestovat. Abych měl přehled, jestli některá část skriptu není například chybně zapsaná, neobsahuje překlepy atd., při jeho spuštění nechám na obrazovku vypsat jednoduchý text „Loading iptables settings“ a za každou jednotlivou část skriptu nechám vypsat na obrazovku tečku. Má to informativní účel pro případ, když by skript vypsal chybové hlášení, tak ať máme přehled, v která části to podle počtu vypsaných teček bylo. Navíc je tento zápis vhodný pro situaci, kdy je skript umístěn do bootu OS Debian, aby uživatel měl přehled, že se v danou chvíli

načítají nastavení iptables. Konec skriptu je potvrzen vypsáním slova „done“ Počáteční hlášení, tečky a ukončení zapíšeme:

```
echo
echo -n "Loading iptables settings"
echo -n "."
echo done.
exit
```

Nyní jsme tedy připraveni ke spuštění skriptu. Jeho funkčnost ověříme tak, že si na virtuálním stroji OS Windows, který představuje vnitřní síť, spustíme jednu ze služeb, která je v síti umístěna. Nejjednoduší je asi zvolit službu typu FTP. Nyní mimo virtuální stroje spustíme libovolného FTP klienta, pomocí kterého se budeme připojovat na veřejnou adresu naší virtuální sítě. Tato simulace představuje situaci, kdy se z vnějšího prostředí Internetu připojuje neznámý uživatel.

Na virtuálním stroji jsem založil FTP server pomocí bezplatného softwaru Star FTP server 1.10. Připojení k serveru a následovné stažení testovacího souboru proběhlo bez problémů.



Obr. č. 14. – Ověření FTP spojení

Úspěšné navázání spojení je nám důkazem, že jsme vše nastavili správně.

6.9 Zařazení skriptu do bootu systému Debian

Jelikož se nastavení iptables při každém spuštění systému vrací do původních hodnot, je třeba náš skript zařadit mezi ostatní úlohy, které se při načítání systému vykonávají.

Skript zkopírujeme do složky /etc/init.d/, čímž se zařadí mezi ostatní skripty spouštěné při startu. Abychom potvrdili zapsání našeho scriptu do bootu systému, musíme použít příkaz [13]:

```
update-rc.d skript defaults
```

```
debian:~# update-rc.d skript defaults
update-rc.d: warning: /etc/init.d/skript missing LSB information
update-rc.d: see <http://wiki.debian.org/LSBInitScripts>
 Adding system startup for /etc/init.d/skript ...
   /etc/rc0.d/K20skript -> ../init.d/skript
   /etc/rc1.d/K20skript -> ../init.d/skript
   /etc/rc6.d/K20skript -> ../init.d/skript
   /etc/rc2.d/S20skript -> ../init.d/skript
   /etc/rc3.d/S20skript -> ../init.d/skript
   /etc/rc4.d/S20skript -> ../init.d/skript
   /etc/rc5.d/S20skript -> ../init.d/skript
debian:~# _
```

Obr. č. 15 – Zápis skriptu do spouštění systému

Na obrázku č. 15 vidíme, co se nám vypíše po vykonání příkazu. Varování upozorňuje na to, že skript není zapsán přesně dle konvencí stanovených specifikací **Linux Standard Base Core Specification 3.1.** Více se o této problematice můžeme dozvědet na internetových stránkách, které nám byly vypsány. Nicméně zápis, jakým máme skript napsán, ničemu nevadí a kdybychom chtěli skript upravit přesně podle stanovené specifikace, podstatně by to zvýšilo délku skriptu.

Po vypsání varování je již skript zapisován do jednotlivých úrovní. OS Debian má několik úrovní, tzv. run-levelů, podle kterých určuje, které procesy se budou v daném módu systému spouštět. Správnost postupu ověříme znovuspuštěním operačního sytému.

```
Mounting local filesystems...done.
Activating swapfile swap...done.
Setting up networking....
Configuring network interfaces...done.
Starting portmap daemon....
Starting NFS common utilities: statd.
Setting console screen modes and fonts.
INIT: Entering runlevel: 2
Starting enhanced syslogd: rsyslogd.
Starting ACPI services....
Starting OpenBSD Secure Shell server: sshd.
Starting MTA: exim4.

Loading iptables settings......done.
```

Obr. č. 15 – Ukázka části spouštění systému s již zavedeným skriptem

7 Zhodnocení

Cílem prvních kapitol práce byla analýza a vyhodnocení modelu TCP/IP, jeho jednotlivých vrstev a jejich významu pro zabezpečení. Z těchto informací jsem těžil ve třetí a čtvrté kapitole, kde jsem se zabýval zabezpečovacími síťovými prvky a vhodnou volbou síťové topologie, především z hlediska ochrany. Výstupem těchto dvou částí je návrh vhodné počítačové sítě pro středně velkou organizaci v kapitole páté.

V předloženém návrhu se podařilo skloubit několik důležitých vlastností počítačové sítě. Síť je přehledně rozčleněna do tří segmentů, což usnadňuje její konfiguraci a zefektivňuje implementované bezpečnostní opatření. To se výrazně promítá i do celkových nákladů potřebných k realizaci sítě. Prvky zabezpečení jsou umístěny tak, aby byly efektivně využity a jejich použití nebylo příliš komplikované či dokonce aby zbytečně neomezovalo uživatele sítě. Správa a konfigurace prvků zabezpečení je prováděna z jednoho bodu, takže není náročná na lidské zdroje. Zpracování a ochrana citlivých dat je zajištěna kombinací více postupů a potenciální útočník by musel překonat několik obranných linií, aby se k chráněným informacím dostal. Síť je chráněná především proti útokům zvenčí (z Internetu), ale připravena je i na útoky zevnitř (od zaměstnanců). Veškerá činnost v síti je sledována a jakákoliv podezřelá aktivita je hlášena příslušné autoritě. Komunikace mezi jednotlivými segmenty sítě je omezena jen na výměnu nezbytných informací, čímž je značně omezena možnost útoku na této úrovni. Bezpečnostní opatření nijak neomezují běžné potřeby dané organizace jako je provoz webových stránek (včetně internetového obchodu), e-mailové služby a přístup zaměstnanců na Internet.

V předloženém návrhu sítě jsme kladli velký důraz na zabezpečení a ochranu tak, jak nám bylo stanoveno v zadání bakalářské práce. Z tohoto důvodu je navrhnutá síť obsáhlejší, než tomu je zvykem u bežných středně velkých sítí. Přestože lze návrh zjednodušit umístěním více služeb na jednu stanici nebo server, pro samotnou praktickou realizaci byl návrh příliš komplexní. Vzhledem k náročnosti praktické implementace všech prvků návrhu, jsme museli upustit od realizace celé sítě a zaměřit se na hraniční směrovač s firewallem.

Na uskutečnení směrovačem s firewallem jsem pracoval v šesté, poslední kapitole. Realizace proběhla pomocí OpenSource softwaru GNU/Linux Debian. Pro snazší konfiguraci a lepší efektivitu práce byla k realizaci a testování použita virtualizace operačních systémů.

Výsledkem je router s firewallem, který odděluje vnitřní privátní síť od vnějšího prostředí Internetu. Jeho úkolem je zároveň chránit vnitřní síť před nežádoucími stavy, útoky či pokusy o vniknutí. Tento prvek zároveň řídí síťový provoz na rozhraní mezi vnitřní sítí a Internetem.

Jeho nasazení je určeno pro situaci, kdy ve vnitřní síti máme umístěny služby typu FTP, SSH, SMTP a WWW, které mají být k dispozici pro uživatele vnější sítě Internetu. Prvek byl nakonfigurován tak, aby splňoval hlavní požadavky na zabezpečení stanovené v teoretické části, kde byl především kladen důraz na zabezpečení.

Literatura

[1] HOŘČICA, Adam. Počítačové sítě [online]. 2004 , 25.11.2004 [cit. 2008-11-02]. Dostupný z WWW: <http://emp.wz.cz/net/main.php?side=obsah>.

[2] DOSTÁLEK, Libor, et al. Velký průvodce protokoly TCP/IP: bezpečnost. [s.l.] : [s.n.], 2003. 592 s. ISBN 807226849X.

[3] RFC 791 – Internet Protocol [online]. 1981 [cit. 2008-11-05]. Dostupný z WWW: <http://tools.ietf.org/html/rfc791>.

[4] RFC793 - Transmission Control Protocol [online]. 1981 [cit. 2008-11-05]. Dostupný z WWW: <http://www.faqs.org/rfcs/rfc793.html>.

[5] RFC 768 - User Datagram Protocol [online]. 1980 [cit. 2008-11-06]. Dostupný z WWW: <http://tools.ietf.org/html/rfc768>.

[6] NORTHCUTT, Stephen, et al. Bezpečnost počítačových sítí. [s.l.] : [s.n.], 2005. 592 s. ISBN 8025106977.

[7] KOŠŤÁL , David, STAUDEK, Jan. Firewally, bezpečnostní oddělovací uzly. [s.l.] : [s.n.], 1999. 12 s.

[8] HAJNÝ, Jan. Návrh řešení autentizace uživatelů pro malé a střední počítačové sítě. [s.l.], 2008. 65 s. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ. Diplomová práce.

[9] BOTOŠ , Csaba. *Root.cz : informace nejen ze světa Linuxu.* [online]. 10. 1. 2006 [cit. 2009-05-05]. Vše o iptables. Dostupný z WWW: <http://www.root.cz/clanky/vse-o-iptables-uvod/>. ISSN 1212-8309.

[10] MALANÍK, Jan. Debian /etc/network/interfaces - konfigurace síťových rozhraní. *AbcLinuxu* [online]. 2007 [cit. 2009-05-13]. Dostupný z WWW:

<http://www.abclinuxu.cz/clanky/system/debian-etc-network-interfaces-konfigurace-sitovych-rozhrani>.

[11] PETŘÍČEK , Miroslav. *Root.cz : informace nejen ze světa Linuxu.* [online]. 18.12.2001. 2006 [cit. 2009-05-05]. Stavíme firewall . Dostupný z WWW: <http://www.root.cz/clanky/stavime-firewall-1/>. ISSN 1212-8309.

[12] VEPSTAS , Linas . Linux Network Address Translation [online]. c1996 , 2002 [cit. 2009-05-06]. Dostupný z WWW: <http://linas.org/linux/load.html>.

[13] KEMP, Steve. *Making scripts run at boot time with Debian* [online]. 2004 [cit. 2004-11-11]. Dostupný z WWW: <http://www.debian-administration.org/articles/28>.

Seznam zkratek

FTP File Transfer Protocol

SMTP Simple Mail Transfer Protocol

DMZ Demilitarizovaná Zóna

ISO International Organization for Standardization

HTTP Hypertext Transfer Protocol

DNS Domain name service

DNAT Destination NAT

NAT Network Address Translation

OS Operační Systém

OSI Open Systems Interconnection

SNAT Source NAT

TCP Transmission Control Protocol

VPN Virtual Private Network

IP Internet Protocol

Seznam příloh